Università Politecnica delle Marche

FACOLTÀ DI INGEGNERIA
Corso di Laurea Triennale in Ingegneria Informatica e dell'Automazione

Tesi di laurea triennale

Modellazione e controllo di un drone con un approccio "Data Driven"

Modeling and control of a drone using a "Data Driven" approach

Candidato:
Martina Rossi
Matricola 1081546

Relatore:
Giuseppe Orlando
Correlatore:
Gianluca Ippoliti

Anno Accademico 2019/2020

Ringraziamenti

Per cominciare, vorrei ringraziare il professor Orlando e il professor Ippoliti per avermi guidato in questo periodo di tirocinio e di stesura della tesi.

Un immenso grazie va a mia madre e a mio padre che hanno fatto di tutto per rendere il più piacevole possibile questo percorso universitario, accettando qualsiasi mia scelta, ma soprattutto supportandomi e sopportandomi quando nessun altro era in grado di farlo.

Grazie a nonno, nonna, zia Nadia e Stefano per avermi pensato tutti i giorni, per avermi sempre incoraggiato e per aver sempre creduto in me.

Grazie al mio ragazzo Marco per essere stato il mio punto fisso, per avermi sempre aiutata, spronata a fare meglio e a dare il meglio di me. Grazie per essere sempre stato al mio fianco e per aver sempre avuto una parola di conforto nei periodi difficili. Grazie perché so che su di te potrò sempre contare.

Grazie a Nadia, Maurizio, Alessia, Claude e Teo per avermi trattata e accolta come se fossi della famiglia da sempre.

Grazie a Paola, Serena e Alessia per essere state delle coinquiline fantastiche, per avermi fatto ridere e per avermi fatto compagnia in questi anni; siete state le sorelle che non ho mai avuto.

Grazie alle mie amiche da una vita e agli amici che ho acquisito nel tempo, che ci sono sempre stati per condividere i momenti di svago, ma anche i momenti di tensione.

Grazie ad Asya e Natalia che, oltre ad essere state delle compagne di corso, sono state compagne di studio e di vita, senza le quali l'università non sarebbe stata la stessa.

Il ringraziamento più grande però va a me per non aver mai mollato.

Sommario

In primo luogo, quest'elaborato espone la differenza tra la teoria di controllo Model-Based Control e la teoria di controllo Data-Driven Control. A seguire, vengono descritti i due metodi di controllo adattativi e completamente data driven: CFDL-MFAC e PFDL-MFAC per sistemi non lineari a tempo discreto e verrà fornito un esempio di applicazione di tali metodi per il modello di un pendolo. In secondo luogo, viene mostrata la modellazione matematica di un drone e, essendo lo scopo quello di applicare i metodi di cui sopra ad un drone e in particolare all'angolo di pitch, è stata ricavata la funzione di trasferimento del sottosistema ad esso inerente. Infine, sono mostrate le simulazioni, ricreate nel Simulink di Matlab, nelle quali applichiamo i metodi CFDL-MFAC e PFDL-MFAC alla funzione di trasferimento ricavata.

Indice

Elenco delle figure

Introduzione

Negli ultimi anni, la ricerca si è focalizzata in particolar modo sul design di controller per i quadricotteri, in quanto quest'ultimi hanno preso largo campo nell'uso quotidiano, sia per scopi professionali che per scopi ludici.

Al contempo, lo studio sulle teorie e metodologie di controllo si sono soffermate su una particolare categoria definita Data-Driven Control, che consiste nello sfruttare i dati I/O forniti dal sistema di controllo e che non necessita di conoscere il modello matematico del processo.

In questo documento vengono ampiamente discussi entrambi gli argomenti, fornendo le basi per comprendere al meglio la progettazione di controller CFDL-MFAC e PFDL-MFAC, la modellazione matematica e la dinamica di un quadricottero, con lo scopo di combinarli insieme in una serie di simulazioni nel Simulink del Matlab, analizzando le risposte del sistema ottenuto.

1 Model-Based Control

1.1 Generalità

La teoria Model-Based Control (MBC) è una teoria di controllo moderna, sviluppatasi a seguito del modello spazio-stato introdotto dal Kalman nel 1960. Il suo nome deriva dal fatto che la teoria di controllo moderna è basata sull'esatta conoscenza del sistema controllato. A seguito dello sviluppo di tale teoria di controllo, essa è attualmente utilizzata in molti campi pratici, in particolar modo nel settore aerospaziale. Dall'altra parte però, il settore industriale si è evoluto e conseguentemente anche la tecnologia e i processi ad esso correlati, facendo sì che essi diventassero sempre più complessi.

Come già accennato, attualmente la teoria di controllo MBC è la più utilizzata sia per i sistemi lineari che per i sistemi non lineari. Gli step necessari da effettuare prima di poterla applicare sono i seguenti:

1. Costruzione del modello matematico dell'impianto
2. Progettazione del controllore sulla base del modello matematico ottenuto, con la fiducia che quest'ultimo rappresenti il vero sistema
3. Analisi del controllo ad anello chiuso, facendo sempre riferimento al modello matematico

Notiamo dunque che il modello matematico rappresenta il punto cardine della progettazione e analisi del controller; rappresenta inoltre il criterio di valutazione e il soggetto al quale viene applicata la teoria MBC.

I principali metodi che vengono utilizzati per modellare un impianto sono due:

1. Il metodo dei principi primi, che modella l'impianto controllato secondo delle leggi fisiche o chimiche per poi stabilire dei parametri a seguito di alcuni esperimenti;
2. Il metodo di identificazione del sistema, che sviluppa un modello di impianto di I/O che si trova in un insieme di modelli specificati che coprono il vero sistema, e può approssimare il sistema originale in termini di bias o errore, utilizzando i dati di misurazione online o offline.

Entrambe le modellazioni rappresentano solo un'approssimazione del reale sistema controllato e perciò presentano alcuni errori dovuti ad un'eventuale semplificazione del sistema e dell'ambiente circostante, che generalmente sono complessi; perciò ci saranno sempre delle dinamiche non modellate e incertezze che possono portare dei problemi nell'applicazione pratica come ad esempio una debole robustezza e/o una scarsa sicurezza nel sistema ad anello chiuso.

Sono stati compiuti numerosi esperimenti per poter incorporare la descrizione delle incertezze e degli errori nel modello matematico, col fine di ottenere una teoria del

controllo solida, ma purtroppo le descrizioni di queste incertezze non possono essere ottenute quantitativamente o qualitativamente mediante modelli fisici o modelli di identificazione. Idem per quanto riguarda la quantificazione del limite superiore di incertezza. Sostanzialmente possiamo dedurre che le descrizioni di incertezze ed errori non sono coerenti con i risultati forniti dalla modellazione fisica o dalla modellazione dell'identificazione.

Per poter valutare l'applicabilità di un controllore ad un modello reale, è necessario studiare la stabilità del sistema di controllo nella sua totalità. Una volta ottenuto il criterio di stabilità, esso viene utilizzato per decidere se il sistema ottenuto può essere applicato oppure no. Vi è un legame tra stabilità del sistema con il processo e il controllore:

- Si suppone che il processo sia stabile (ad anello aperto).
- Il modello deve convergere per poter definire stabile il sistema ottenuto; ma è necessario ricordare che il modello converge solo se i segnali che provengono dal circuito sono anch'essi stabili e persistentemente eccitati.

Questo è un circolo vizioso ed è difficile venirne a capo, ma non è possibile definire un altro criterio per valutare la stabilità per sistemi di controllo che dipendono dal modello. Ciò rappresenta uno degli ostacoli principali che incontrano gli MBC. In seguito, invece, vedremo come per gli MFAC si ha un criterio di stabilità generale.

Pertanto, qualora si volesse applicare la teoria MBC è necessario ricordare che essa è difficile da applicare per sistemi della realtà quotidiana e che la sicurezza e la robustezza non sono garantite a cause delle incertezze non modellate. Per questo motivo, il passo più importante per la progettazione di un sistema di controllo con questa tecnica è quello di descrivere il modello matematico nella maniera più accurata possibile cercando di comprendere le possibili anomalie, nonostante sia impossibile ottenere il modello perfetto a causa di limitazioni pratiche e teoriche. Infatti, non è ancora stata scoperta nemmeno una teoria che ci consente di ottenere un perfetto modello matematico del sistema originale. La stessa considerazione vale per i sistemi non lineari complessi che ci sono nella vita reale. Talvolta è addirittura più difficile ottenere una modellazione accurata piuttosto che sintetizzare un controllore accurato. Tra l'altro, maggiore sarà l'accuratezza del modello e maggiori saranno lo sforzo di controllo e la complessità del controllore, i quali comporteranno difficoltà nella gestione del sistema stesso. Inoltre, senza gli input costantemente eccitati è difficile ottenere un modello accurato e tantomeno la convergenza e la stabilità del sistema di controllo ad anello chiuso. "Il metodo di controllo MBC proviene dal modello di sistema e finisce nel modello di sistema."

2 Data-Driven Control

2.1 Generalità

A seguito dello sviluppo e dell'evoluzione scientifica e tecnologica, in particolar modo riguardo l'ambito dell'informazione, molti processi industriali hanno subito dei cambiamenti drastici e repentini; le aziende, per poter stare al passo coi tempi, si sono ingrandite e adattate alla moderna tecnologia e alla complessità degli attuali processi, cercando comunque di mantenere un'alta qualità dei prodotti. È, infatti, difficile utilizzare il metodo dei principi primi o il metodo di identificazione, concludendo dunque che utilizzare la teoria MBC in questo ambito è controproducente. Inoltre, le moderne applicazioni industriali producono e memorizzano innumerevoli quantità di dati di una certa importanza, ottenuti sia in maniera attiva che in maniera passiva, riguardanti lo stato del processo e lo stato dei macchinari utilizzati nel processo industriale. Tali dati acquisiti posso essere sfruttati col fine di creare controller e la successiva analisi laddove il modello del processo non è noto o non disponibile. Da questa osservazione deriva il nome Data-Driven Control (DDC). Inoltre, gli stessi dati potrebbero consentirci di prevedere e valutare gli stati futuri del sistema, le sue prestazioni, i possibili guasti e addirittura di prendere delle decisioni. Il DDC possiede diverse caratteristiche intrinseche quali:

1. Non include né parte né l'intero modello del processo. Esso non è più dipendente dal modello. Infatti, la progettazione di controller di questo tipo non si basa più su questo.
2. Le conclusioni di stabilità e convergenza degli approcci DDC non sono legati alla fedeltà di riproduzione del modello, a meno che esso non faccia uso di informazione di sistema e delle sue dinamiche in maniera implicita (es. controllo adattativo diretto, controllo adattativo sub-spaziale, …).
3. A differenza della teoria MBC, nella teoria DDC non vi sono più problemi di dinamiche non modellate con conseguente scarsa robustezza, in quanto il modello non è conosciuto a prescindere.

Lo sviluppo e l'evoluzione della teoria DDC è di particolare importanza per la teoria del controllo. Finora, infatti, esistono molteplici metodi DDC: il più conosciuto è sicuramente il Proportion Integral Differential (PID), seguito dal Model-Free Adaptive Control (MFAC), Iterative Feedback Tuning (IFT), Virtual Reference Feedback Tuning (VRFT), Iterative Learning Control (ILC), e molti altri. In questa tesi, analizzeremo nel dettaglio soltanto la tipologia MFAC. Nonostante ciò, la teoria DDC è ancora piuttosto giovane come metodo di controllo ed è ancora soggetta a numerosi studi.

La relazione tra metodi di controllo e oggetti controllati è illustrata in figura:

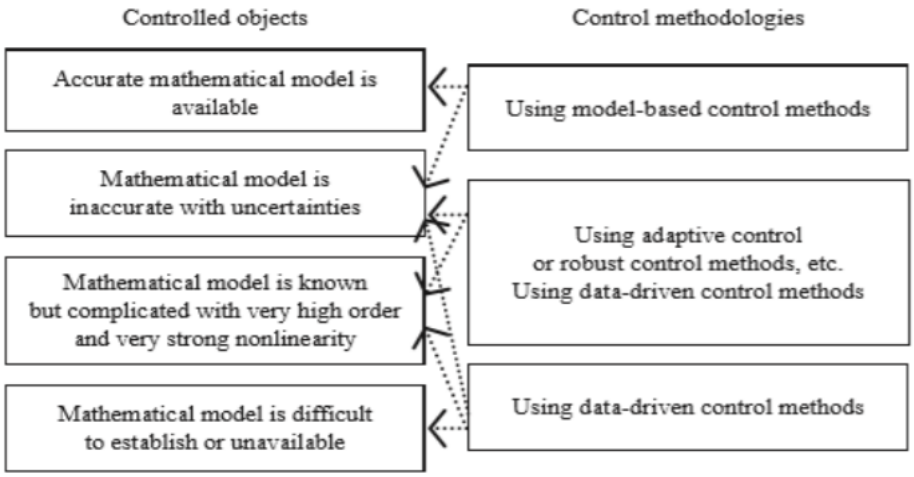

Figura n.1: Oggetti controllati da DDC

Una teoria di controllo ideale dovrebbe essere completa, ovvero dovrebbe poter essere applicabile a tutte le categorie di oggetti di controllo esistenti mostrate in Figura n.1. Da ciò, ne deduciamo che la tecnica di controllo perfetta dovrebbe essere la combinazione dell'MBC e del DDC. Infatti, ricordiamo che il metodo MBC non è in grado di controllare processi sconosciuti e che l'eccessiva specificità nella modellazione del processo controllato (anche non lineare) può portarci a lavorare con ordini elevati e con un controllore piuttosto complesso; pertanto è necessario, talvolta, considerare una semplificazione tramite MBC. Tra l'altro, per implementare un controllo robusto, il limite superiore delle incertezze deve essere fornito quantitativamente e descritto a priori; anche se, difficilmente può essere raggiunto in teoria dagli approcci di modellazione esistenti.

Entrambi i metodi DDC e MBC si propongono di progettare un controllore col fine di pilotare l'output di controllo in modo tale da soddisfare una specifica richiesta; ma entrambi presentano dei pro e dei contro. Sulla base di questa osservazione, i metodi DDC e MBC potrebbero svilupparsi separatamente e lavorare insieme in modo complementare. La differenza principale tra MBC e DDC è che il primo è l'approccio di progettazione del controllo basato su modello laddove è possibile riprodurre fedelmente il processo reale e necessita di strumenti di progettazione e analisi sistematici, il secondo è l'approccio di progettazione del controllo basato sui dati laddove non è possibile riprodurre fedelmente il processo reale o se non è possibile dedurlo. In quest'ultimo, la dipendenza dal modello matematico di un impianto controllato viene rimossa per la progettazione del sistema di controllo e ciò rappresenta il principale vantaggio della teoria e dei metodi DDC. Alcuni problemi importanti che sono inevitabili nei metodi MBC tradizionali, tra cui dinamiche non modellate contro robustezza, modellazione accurata contro semplificazione del modello o riduzione del controller, progettazione del controllo robusta rispetto alla descrizione non disponibile sul limite superiore delle

incertezze, buoni risultati teorici contro cattive prestazioni pratiche e così via, non esistono più nell'ambito del DDC.

3 Model-Free Adaptive Control

3.1 Generalità

"Nella tecnica dei controlli automatici si parla di *controllo adattativo* quando gli interventi sul sistema controllato sono effettuati tenendo conto non solo dello stato del sistema, ma anche di tutti i parametri che caratterizzano l'efficacia del controllo." (Vocabolario online Treccani, Def. 2)

Nel 1994, Zhongsheng Hou propose per la prima volta questa tecnica di controllo MFAC. Decise poi in seguito di applicarla a sistemi non lineari tempo discreto dove, al posto di ricercare il modello non lineare del processo che si vuol controllare, viene identificato un modello lineare equivalente ad ogni operazione del sistema in anello chiuso lungo i punti di funzionamento dinamico. Questa operazione può essere effettuata servendosi del Pseudo Partial Derivative (PPD), un nuovo concetto di una nuova tecnica di linearizzazione dinamica. Esso è una variabile nel tempo e la sua stima viene ricavata acquisendo dati I/O provenienti dal processo controllato.

Le tecniche di linearizzazione dinamica che appartengono a questa categoria di controllo sono: il Compact-Form Dynamic Linearization (CFDL), il Partial-Form Dynamic Linearization (PFDL), e il Full-Form Dynamic Linearization (FFDL).

L'MFAC non si basa né sul modello né su delle regole, ma su delle informazioni. Esso è volto ad ottenere una soluzione semplice, rapida ed efficace. È possibile affermare che le sole informazioni o i dati sfruttati rappresentano un equivalente del modello.

Il MFAC viene definito controller per scopi generici e infatti può essere utilizzato in numerose applicazioni grazie ai suoi pregi: primo tra i quali è quello di essere un modello di controllo dipendente esclusivamente dalle informazioni ottenute in tempo reale (implicando un controller che può essere ottenuto in maniera indipendente); secondo è quello di non necessitare alcun un segnale di test esterno o alcun processo di addestramento (implicando un controller a basso costo); e terzo, MFAC è semplicemente implementabile senza necessitare di un elevato carico computazionale. Inoltre -quarto- sotto alcune ipotesi, è possibile garantire la convergenza monotona, la stabilità BIBO. Quest'ultimo punto verrà verificato nei teoremi a seguire.

Per poter utilizzare tale tecnica di controllo MFAC, l'user può selezionare la categoria più coerente al sistema che si vuole controllare tra quelle descritte sopra, aggiustare i parametri, inizializzare il controllore. Tutto ciò viene fatto per poterne trarre dei vantaggi e per poter evitare una possibile manutenzione.

Lo scopo di tale tecnica è quella di creare un sistema di controllo iterativo, ovvero che sfrutti i dati I/O ottenuti negli istanti passati. Ad ogni passo iterativo, non è necessario modificare i parametri prescelti, ma è necessario che le limitazioni imposte siano sempre rispettate. Inoltre, esso non richiede una regolazione manuale, ma essi si adattano alla nuova dinamica del processo, dei carichi e dei disturbi. Possiamo quindi concludere che si può applicare proprio il metodo MFAC come controllo adattativo per un sistema non lineare con parametri e struttura varianti nel tempo, senza utilizzare né il modello dinamico né le informazioni sulla struttura dell'impianto.

3.2 Linearizzazione per sistemi non lineari a tempo discreto

È necessario sottolineare che la maggior parte dei sistemi che si trovano in natura sono non lineari e dipendenti da numerose variabili e che i sistemi lineari, generalmente, ne rappresentano solo un'approssimazione. Per questo motivo, la ricerca sta cercando di sviluppare analisi e teorie di controllo per sistemi non lineari. Il primo pensiero è stato quello di trasformare il sistema non lineare in uno lineare equivalente. A seguire, vengono presentate alcune delle tipologie di linearizzazione più diffuse come: linearizzazione con feedback, linearizzazione di Taylor, linearizzazione a tratti e linearizzazione basata sull'approssimazione della funzione ortogonale.

Ciascuno di questi metodi ha un proprio limite. Ad esempio: la linearizzazione feedback richiede un modello matematico piuttosto fedele del processo che si vuole controllare; la linearizzazione di Taylor tende ad omettere i termini di ordine elevato generando un modello matematico fin troppo approssimativo; per la linearizzazione a tratti sono necessarie molte informazioni, come l'istante di commutazione e il tempo di sosta della dinamica linearizzata a tratti di un impianto controllato; la linearizzazione ortogonale comporta un elevato numero di parametri, il che porta a un pesante carico computazionale per gli algoritmi di identificazione dei parametri e la complessità nella progettazione del controller.

L'utilizzo dei metodi di linearizzazione descritti sopra richiede un'accurata descrizione dinamica del modello dell'impianto oppure determina un'influenza più o meno negativa sulla progettazione del controllore o sull'analisi del sistema. In altre parole, questi metodi di linearizzazione non sono particolarmente adatti alla progettazione del controllo. Una linearizzazione orientata alla progettazione del controllo dovrebbe avere caratteristiche come: una struttura semplice, una quantità

moderata di parametri regolabili, un comodo utilizzo dei dati di I/O direttamente durante la progettazione del controllore e così via.

Come abbiamo già accennato in precedenza, esiste una moderna tecnica di linearizzazione dinamica creata appositamente per i sistemi non lineari a tempo discreto basati sul concetto PPD (ma anche per altri concetti di questo tipo, come il PG), che si comporta come un metodo di linearizzazione orientato al progetto di controllo. Con questo approccio, individuiamo una serie di modelli di dati di linearizzazione dinamica sotto forma di un incremento dei dati di I/O: CFDL, PFDL e FFDL. Anche se in questa tesi ci focalizzeremo solo sui primi due metodi.

4 CFDL

Innanzitutto, consideriamo una classe di sistemi SISO non lineari a tempo discreto:

$$y(k+1) = f\left(y(k),\ldots,y\left(k-n_y\right),u(k),\ldots,u(k-n_u)\right) \quad (1)$$

Dove u (k) ∈ R sono gli input e y (k) ∈ R sono gli output del controllore all'istante k, n_y e n_u sono due numeri interi positivi non noti che rappresentano, rispettivamente, gli ordini degli output e degli input, mentre $(\ldots)$: $R^{n_u+n_y+2} \to R$ è una funzione non lineare anch'essa non nota.

Ora consideriamo le seguenti assunzioni:

- Assunzione A: La derivata parziale di $f(\ldots)$ rispetto alla variabile $(n_y + 2) - esima$ è continua ∀ k.
- Assunzione B: Il sistema (1) sopra descritto soddisfa la condizione

  generalizzata di Lipschitz ∀ k:

  $$|y(k_1+1) - y(k_2+1)| \leq b\,|u(k_1) - u(k_2)|$$

  con $u(k_1) \neq u(k_2)$ per $k_1 \neq k_2 \wedge k_1, k_2 \geq 0$, dove

  $$y(k_i+1) = f\left(y(k_i),\ldots,y\left(k_i-n_y\right),u(k_i),\ldots,u(k_i-n_u)\right)$$

  con $i = 1{,}2 \ \wedge b > 0$.

Le assunzioni A e B, definite sopra, hanno senso per il processo.

- L'assunzione A è una restrizione tipica fatta per i sistemi non lineari all'interno del controllo.

- L'assunzione B impone un limite superiore alla velocità con cui l'uscita del sistema controllato varia dal suo ingresso.

Energeticamente parlando, se la variazione di energia dell'ingresso di controllo è finita allora la variazione di energia interna del sistema non può tendere a infinito.

A breve utilizzeremo le seguenti notazioni: $\Delta y(k+1) = y(k+1) - y(k)$ e $\Delta u(k) = u(k) - u(k-1)$ che rappresentano rispettivamente la variazione di uscita e la variazione di ingresso tra due istanti consecutivi.

Ora è dunque possibile definire il CFDL.

Teorema 1: Considerando il sistema (1) che soddisfi le assunzioni A e B. Se $|\Delta u(k)| \neq 0$ allora esiste un parametro variante nel tempo $\varphi_c(k) \in R$, detto Pseudo Partial Derivative (PPD) tc il sistema (1) può essere riscritto nel seguente modello CFDL:

$$\Delta y(k+1) = \varphi_c(k)\Delta u(k) \qquad (2)$$

dove $\varphi_c(k)$ limitato ∀ k.

Da notare la dipendenza dal tempo del parametro PPD anche se il sistema considerato non dipende dal tempo. In particolare, notiamo che il PPD presenta un legame con i valori assunti dai segnali di input e dai segnali di output dagli istanti precedenti fino all'istante presente.

$\varphi_c(k)$ comprende gli indici di tutti i valori fino all'istante k, ma se $\Delta u(k)$ e il periodo di campionamento sono sufficientemente piccoli, allora esso può essere considerato come un parametro lentamente variante nel tempo. La stima del suo comportamento numerico può essere facilmente ottenuta.

Per un sistema non lineare semplice di questo tipo: $y(k+1) = f\big(u(k)\big)$, PPD rappresenta la derivata della funzione non lineare $f(\cdot)$ in un punto compreso tra $u(k+1)\ e\ u(k)$.

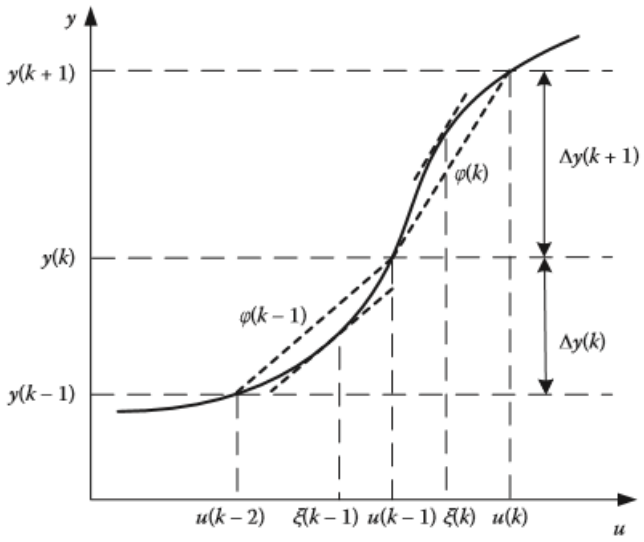

Figura n.2: PPD geometricamente rappresentato

Il modello linearizzato (2) viene calcolato periodicamente - ad ogni istante di campionamento - risulta essere fedele al modello reale e non varia in maniera repentina. Tale metodo non può essere utilizzato per altri scopi al di fuori del design del controllore.

Se il sistema (1) , se $\varphi_c(k)$ non è nullo o se $\varphi_c(k)$ non tende all'infinito, è controllabile in uscita per uno specifico punto scelto qualsiasi sia k. Viene appunto sfruttato $\varphi_c(k)$ per valutare se un sistema può essere controllato in uscita e viene stimato solo a partire dalle informazioni acquisite fino all'istante corrente; non può però essere determinato analiticamente perché la sua limitatezza e diversità da zero, solitamente, possono essere verificate sfruttando le informazioni raccolte nel sistema a ciclo chiuso.

Dal Teorema 1 notiamo che la limitatezza di PPD $\varphi_c(k)$ deriva dall'assunzione B fatta sul sistema. Il Teorema 1 sottolinea che si deve avere $|\Delta u(k)| \neq 0 \; \forall$ k; se, al contrario, dovesse accadere che $|\Delta u(k)| = 0$ allora dovremmo applicare una linearizzazione leggermente diversa dopo essersi spostati di un valore $\sigma_k \in Z$ istanti di tempo fino a $|u(k)| \neq u(k - \sigma_k)$.

Teorema 2: Per sistemi non lineari del tipo (1), dopo aver verificato che siano soddisfatte le assunzioni A e B, qualora esistesse un intero $k_0 \geq 1$ tc:

$$\Delta u(j) = 0 \; se \; j = 1, \dots, k_0 - 1$$

$$o$$

$$\Delta u(j) \neq 0 \; se \; j = \; k_0$$

Allora $\forall k \geq k_0$, si può sempre trovare un intero finito σ_k tc:

$$\Delta u(k-j) = 0 \; se \; j = 1, \dots, \sigma_k - 2$$

$$o$$

$$\Delta u(k-j) \neq 0 \; se \; j = \; \sigma_k - 1$$

Contemporaneamente, esiste un PPD $\varphi_c(k)$ tc il sistema (1) può essere trasformato nel seguente modello CFDL:

$$y(k+1) - y(k-\sigma_k+1) = \varphi_c(k)(u(k) - u(k-\sigma_k)$$

Con il parametro $\varphi_c(k) \; \leq b \; \forall$ k, che è uno scalare che racchiude tutti i possibili comportamenti complessi e indesiderati come ad esempio la non linearità, parametri varianti nel tempo e molto altro riguardo al sistema originale. Essenzialmente, questi comportamenti complicati sono fusi e inseriti all'interno di $\varphi_c(k)$, rendendolo difficile da stimare.

5 PFDL

Il PFDL è un altro metodo di linearizzazione e può essere considerato un miglioramento del CFDL. Esso considera l'influenza che l'input di controllo – il quale possiede una certa lunghezza prestabilita - ha sull'output di controllo negli istanti futuri. In questo caso, le anomalie che possono essere riscontrate nel comportamento del sistema originale vengono gestite facendo uso di un maggior numero di parametri.

Viene definito $U_L(k) \in R^L$ il vettore di tutti i segnali di controllo nella finestra mobile di dimensioni [k – L+ 1, k]: $U_{\,L}(k) = [\,u(k), \dots, u(k-L+1)]^T$ dove $U_L(k) = 0$ in corrispondenza di k ≤ 0. L è un numero intero definito come Linearization Lenght Constant (LLC). In particolar modo, possiamo notare come il modello di dati PFDL coincida con quello CFDL nel caso in cui L=1. Si possono avere diversi modelli di dati PFDL in base alle diverse lunghezze dell'input di controllo LLC L.

Anche qui per poter definire il PFDL è necessario considerare alcune assunzioni:

- Assunzione C: Le derivate parziali di $f(\,.\,.\,.)$ rispetto alle variabili dalle $(n_y+2) - esime$ alla $(n_y + L + 1\,)$ – ovvero $u(k), \dots, u(k-L+1)$ – sono continue ∀ k.
- Assunzione D: Il sistema (1) soddisfa la condizione generalizzata di Lipschitz ∀ k:

  $$|y(k_1+1) - y(k_2+1)| \leq b\,|U_L(k_1) - U_L(k_2)|$$

  Per $U_L(k_1) \neq U_L(k_2)$ e con $k_1 \neq \; k_2, k_1, k_2 \geq 0$,

dove $y(k_i + 1) = f(y(k_i), \dots, y(k_i - n_y),$

$u(k_i), \dots, u(k_i - n_y))\ \ i = 1{,}2\ \wedge b > 0$

Dopo aver posto $\Delta U_L(k) = U_L(k) - U_L(k-1)$ possiamo proporre il teorema del metodo PFDL sul sistema (1).

Teorema 3: Consideriamo il sistema non lineare (1), dopo aver verificato che siano soddisfatte le assunzioni C e D. ∀ L fissato, se $|\Delta U_L(k)| \neq 0$, allora esiste un vettore, variante nel tempo, $\varphi_{p,L}(k) \in R^L$, detto Pseudo Gradient (PG) tc il sistema (1) possa essere trasformato nel seguente modello dati PFDL:

$$\Delta y(k+1) = \varphi_{p,L}^T(k)\Delta U_L(k)$$

Con $\varphi_{p,L}^T(k) = [\varphi_1(k), \dots\ , \varphi_L(k)]^T$ per ogni ∀ k.

Le dimensioni di PG nel modello di dati PFDL sono maggiori rispetto a quelle del modello CFDL, mentre le dinamiche di PG nel modello PFDL sono meno complesse rispetto a quelle di PPD del modello CFDL.

Da queste osservazioni, ne consegue che $\varphi_{p,L}(k)$ può essere facilmente stimato tramite l'algoritmo utilizzando le informazioni del sistema a ciclo chiuso. Questo particolare approccio che ci consente di scrivere il sistema non lineare originale in uno dinamico linearizzato equivalente, il quale può diventare più flessibile scegliendo una dimensione di PG e di L appropriata.

6 MFAC per sistemi SISO non lineari

6.1 CFDL basato su MFAC

Consideriamo una classe di sistemi SISO a tempo discreto non lineari come segue:

$$y(k+1) = f\left(y(k),\dots,y(k-n_y),u(k),\dots,u(k-n_u)\right) \qquad (3)$$

Dove u (k) ∈ R sono gli input e y (k) ∈ R sono gli output del controllore all'istante k, n_y e n_u sono due numeri interi positivi non noti che rappresentano, rispettivamente, gli ordini degli output e degli input, mentre (. . .): $R^{n_u+n_y+2} \to R$ è una funzione non lineare anch'essa non nota.

Dal Teorema 1, notiamo che il sistema soddisfa le assunzioni A e B con $|\Delta u(k)| \neq 0$ ∀ k. Quindi è possibile trasformarlo nel metodo di dati CFDL:

$$\Delta y(k+1) = \varphi_c(k)\Delta u(k) \qquad (4)$$

Dove $\varphi_c(k) \in R$ è il limite del sistema (3).

L'equazione (4) rappresenta quindi la linearizzazione dinamica equivalente del sistema non lineare. Lo schema di controllo di questa tecnica si baserà su questa linearizzazione (4) - scritta sottoforma incrementale - del modello fornito, il quale è costituito da dati variabili nel tempo e il parametro scalare $\varphi_c(k)$.

6.1.1 Algoritmo di controllo

$$J\big(u(k)\big) = |y^*(k+1) - y(k+1)|^2 + \lambda|u(k) - u(k-1)|^2 \qquad (5)$$

L'equazione (5) è la funzione che viene usata per definire l'algoritmo del controllore. Viene scelto $\lambda > 0$, che è un parametro che introduce delle limitazioni sulla variazione dell'input di controllo. Se sostituiamo il modello CFDL (3) nella funzione di costo (4), differenziando rispetto a $u(k)$, dall'equazione (5) otteniamo la seguente:

$$u(k) = u(k-1) + \frac{\rho\, \varphi_c(k)}{\lambda + |\varphi_c(k)|^2}\left(y^*(k+1) - y(k)\right) \qquad (6)$$

L'equazione (6) ottenuta è una legge di controllo. Viene scelto $\rho \in (0,1]$, il quale è un parametro introdotto per generalizzare maggiormente l'algoritmo di controllo.

Aggiungendo λ in (6) si va a penalizzare $\Delta u(k)$, ovvero la variazione dell'ingresso di controllo, con lo scopo di ottenere un segnale di controllo più "smooth". Allo stesso tempo, però, λ se scelto in maniera opportuna può condurci ad una situazione di stabilità e convergenza. λ rappresenta dunque un parametro con un certo peso per l'MFAC.

Per poter utilizzare la formula (6) è necessario che PPD sia noto, ma dobbiamo ricordare che esso è un parametro tempo variante e che la sua stima accurata è difficile da ottenere quando il modello del sistema non è noto. È necessario dunque definire un parametro variante nel tempo che ci permetta di stimare PPD online a partire dai dati I/O acquisiti dal processo controllato. Per farlo è necessario passare per questo step:

$$J\big(\varphi_c(k)\big) = |y(k) - y(k-1) + \varphi_c(k)\Delta u(k-1)|^2 + \mu|\varphi_c(k)| - \varphi_c(k-1) \qquad (7)$$

La funzione (7) ci consente di stimare PPD in maniera adeguata e viene inserito il parametro $\mu > 0$ che funge da fattore di peso.

Minimizzando (7) rispetto a φ_c si ottiene la seguente equazione o algoritmo di stima di PPD:

$$\widehat{\varphi_c}(k) = \widehat{\varphi_c}(k-1) + \frac{\eta\, \Delta u(k-1)}{\mu + \Delta u(k-1)^2}\left(\Delta y(k) - \widehat{\varphi_c}(k-1)\Delta u(k-1)\right) \qquad (8)$$

Nella quale, $\eta \in (0,2]$ è un parametro inserito per generalizzare maggiormente l'algoritmo e per renderlo ancora più flessibile, mentre la presenza di μ al

denominatore ci impedisce di effettuare la divisione per 0 e consentendoci di determinare un fattore che penalizzi il variare del termine che stima PPD.

$\widehat{\varphi_c}(k)$ è la stima di φ_c.

6.1.2 Schema di controllo

Quando l'algoritmo di controllo (4) viene integrato con il PPD stimato si ottiene il seguente schema di controllo CFDL-MFAC:

$$\widehat{\varphi_c}(k) = \widehat{\varphi_c}(k-1) + \frac{\eta\,\Delta u(k-1)}{\mu+\Delta u(k-1)^2}\left(\Delta y(k) - \widehat{\varphi_c}(k-1)\Delta u(k-1)\right) \quad (9)$$

$$\widehat{\varphi_c}(k) = \widehat{\varphi_c}(1) \;\; se \;\; |\widehat{\varphi_c}(k)| \le \varepsilon \;\; o \;\; |\Delta u(k-1)| \le \varepsilon \;\; o \;\; sign\left(\widehat{\varphi_c}(k)\right) \neq sign\left(\widehat{\varphi_c}(1)\right) \quad (10)$$

$$u(k) = u(k-1) + \frac{\rho\,\varphi_c(k)}{\lambda+|\varphi_c(k)|^2}\left(y^*(k+1) - y(k)\right) \quad (11)$$

Dove $\lambda > 0, \mu > 0,\ \rho \in (0,1],\ \eta \in (0,2]$, $\widehat{\varphi_c}(1)$ è il valore iniziale di $\widehat{\varphi_c}(k)$ e ε è una costante molto piccola positiva.

La condizione (10) consente al parametro di stima di seguire il parametro variante nel tempo in maniera ottimale.

Il PPD $\varphi_c(k)$ non tende a cambiare col variare dei parametri o della struttura o a seguito di ritardi, infatti il CFDL-MFAC è costretto a sfruttare le sue proprietà di adattamento e la sua elevata robustezza che sono rare nel caso di controllo adattativo. Tuttavia, questo non lo rende un metodo applicabile a qualsiasi tipo di sistema non lineare a tempo discreto, ma solo ad una particolare sottoclasse che rispetta alcune assunzioni.

È necessario che siano soddisfatte le seguenti assunzioni per poi studiare e analizzare la stabilità:

- Assunzione E: per una uscita desiderata limitata $y^*(k+1)$, esiste un ingresso di controllo limitato $u^*(k)$ tc l'uscita del sistema guidata da $u^*(k)$ è uguale a $y^*(k+1)$.
- Assunzione F: il segno di PPD si ritiene immutato ∀ k e $\Delta u(k) \neq 0$ che $\varphi_c(k) > \varepsilon$ è soddisfatta, dove ε è una costante molto piccola positiva.

Le assunzioni E e F, definite sopra, hanno senso per il processo.

- L'assunzione E implica che il problema di controllo ha soluzione e che quindi è possibile controllare dall'uscita il sistema.
- L'assunzione F, fisicamente parlando, implica che l'output non può diminuire all'aumentare dell'input di controllo.

Teorema 4: Se il sistema non lineare (3), soddisfa le assunzioni A, B, E e F viste in precedenza, esso viene controllato da uno schema CFDL-MFAC (9) e (11) per risolvere il seguente problema: $y^*(\mathrm{k}+1) = y^* = costante$. Esiste inoltre una costante $\lambda_{min} > 0$ tc sono valide le seguenti proprietà $\forall\, \lambda > \lambda_{min}$:

- L'errore di tracciamento dell'uscita converge in maniera monotona: $\lim_{k\to\infty} |y^*(k+1) - y(k+1)|$
- Il sistema a ciclo chiuso è BIBO stabile – $y(k), u(k)$ limitati.

Abbiamo già evidenziato in precedenza che, nel modello di dati CFDL, se il sistema non lineare è complesso allora lo sarà anche la dinamica del PPD $\varphi_c(k)$. In tal caso, il controllore potrebbe portarci ad instabilità poiché la variazione dell'uscita del sistema all'istante successivo è legata alla variazione dell'ingresso all'istante corrente, all'interno di una finestra mobile.

6.2 PFDL basato su MFAC

Il PFDL-MFAC si basa sul modello di dati PFDL. A differenza del modello di dati CFDL-MFAC, il PFDL-MFAC presenta un più elevato numero di gradi di libertà e una maggiore flessibilità in termini di controllore.

Dal Teorema 4, notiamo che il sistema non lineare (4) soddisfa le assunzioni C e D con $|\Delta U_L(k)| \neq 0\ \ \forall$ k. Quindi è possibile trasformarlo nel metodo di dati PFDL:

$$\Delta y(k+1) = \varphi_{p,L}^T(k)\Delta U_L(k) \qquad (11)$$

Con $\varphi_{p,L}^T(k) = [\varphi_1^T(k), \dots\ , \varphi_L^T(k)\,]^T \in R^L$ che è il PG sconosciuto limitato, mentre $\Delta U_L(k) = [\Delta u(k), \dots\ , \Delta u(k-L+1)]^T$ e L è l'input di controllo LLC.

6.2.1 Algoritmo di controllo

Consideriamo la seguente funzione:

$$J\big(u(k)\big) = |y^*(k+1) - y(k+1)|^2 + \lambda |u(k) - u(k-1)|^2 \qquad (12)$$

(12) rappresenta la funzione per l'ingresso di controllo. Qui viene aggiunto un parametro $\lambda > 0$ che funge da fattore di peso.

Andando a sostituire il modello di dati PFDL (11) all'interno di (12) e minimizzandola rispetto a $u(k)$ ci porta al seguente algoritmo di controllo:

$$u(k) = u(k-1) + \frac{\rho\, \varphi_c(k)}{\lambda + |\varphi_c(k)|^2} \left(y^*(k+1) - y(k)\right) \frac{\varphi_1(k) \sum_{i=2}^{L} \rho_i \varphi_i(k) \Delta u(k-i+1)}{\lambda + |\varphi_1(k)|^2}) \quad (13)$$

Dove $\rho_i \in (0,1]\ con\ i = 1,2,\dots,L$ è stato inserito per migliorare la flessibilità all'algoritmo.

Anche qui è necessario definire un parametro tempo variante che ci permetta di stimare PG online a partire dai dati I/O acquisiti dal processo controllato. Per farlo è necessario passare per questo step:

$$J(\varphi_{p,L}(k)) = |y(k) - y(k-1) - \varphi_{p,L}^T(k)\Delta U(k-1)|^2 + \mu|\varphi_{p,L}(k) - \varphi_{p,L}(k)|^2 \quad (14)$$

Dove viene introdotto il parametro μ>0.

Dopo aver effettuato la minimizzazione di (14) rispetto $\varphi_{p,L}(k)$, in accordo con le condizioni di ottimalità si ottiene:

$$\widehat{\varphi_{p,L}}(k) = \widehat{\varphi_{p,L}}(k-1) + \frac{\eta \Delta U_L(k-1)\left(y(k) - y(k-1) - \widehat{\varphi_{p,L}}^T(k-1)\Delta U_L(k-1)\right)}{\mu + |\Delta U_L(k-1)|^2} \quad (15)$$

In cui $\eta \in (0,2]$ è un parametro che è stato aggiunto per generalizzare maggiormente il tutto.

$\widehat{\varphi_{p,L}}(k)$ è la stima di $\varphi_{p,L}(k)$, che è un vettore tempo variante di dimensione L e non più uno scalare come nello schema CFDL.

6.2.2 Schema di controllo

È possibile definire il PFDL-MFAC nel seguente modo a partire dalle equazioni di stima e del controllore (13) e (15):

$$\widehat{\varphi_{p,L}}(k) = \widehat{\varphi_{p,L}}(k-1) + \frac{\eta \Delta U_L(k-1)\left(y(k) - y(k-1) - \widehat{\varphi_{p,L}}^T(k-1)\Delta U_L(k-1)\right)}{\mu + |\Delta U_L(k-1)|^2} \quad (16)$$

$$\widehat{\varphi_{p,L}}(k) = \widehat{\varphi_{p,L}}(1)\ \ se\ \ |\widehat{\varphi_{p,L}}(k)| \le \varepsilon\ \ o\ \ |\Delta U_L(k-1)| \le \varepsilon\ \ o\ \ sign\left(\widehat{\varphi_1}(k)\right) \ne sign\left(\widehat{\varphi_1}(1)\right) \quad (17)$$

$$u(k) = u(k-1) + \frac{\rho\, \widehat{\varphi_1}(k)}{\lambda + |\widehat{\varphi_1}(k)|^2} \left(y^*(k+1) - y(k)\right) - \frac{\widehat{\varphi_1}(k) \sum_{i=2}^{L} \rho_i \widehat{\varphi_i}(k) \Delta u(k-i+1)}{\lambda + |\widehat{\varphi_1}(k)|^2}) \quad (18)$$

Con $\lambda > 0,\ \mu > 0,\ \eta \in (0,2],\ \rho_i \in (0,1]\ con\ i = 1,2,\dots,L.\varepsilon$ è una contante molto piccola positiva e $\widehat{\varphi_1}(k)$ è il valore iniziale di $\widehat{\varphi_{p,L}}(k)$.

Viene aggiunto un meccanismo di ripristino con lo scopo di stimare i parametri in maniera corretta.

$\widehat{\varphi_{p,L}}(k)$ è il vettore L-dimensionale che ci consente di stimare PG. L può variare da algoritmo e algoritmo, ma deve essere per forza scelto tra un intervallo di valori che parte da 1 e termina con la somma approssimativa dell'ordine del processo qualora n_u e n_y siano sconosciuti e, generalmente, viene scelto tanto più grande quanto più è complesso il sistema, in modo tale da gestirlo meglio. Per L=1, il PFDL-MFAC coincide con il CFDL-MFAC. Infatti, ricordiamo che la principale differenza tra PFDL-MFAC e CFDL-MFAC è data dal maggior numero di parametri da parte del primo rispetto al secondo, dovuti all'aggiunta dei parametri $\rho_1, \rho_2, \dots, \rho_L$ che ci consentono di aumentare i gradi di libertà e la flessibilità.

Teorema 5: Se il sistema non lineare (4), soddisfa le assunzioni A, B, E, e F descritte in precedenza ed è controllato con lo schema PFDL-MFAC in modo tale che $y^*(k+1) = y^* = costante$, allora esiste una costante $\lambda_{min} > 0$ tc sono valide le seguenti proprietà $\forall\ \lambda > \lambda_{min}$:

- L'errore di tracciamento dell'uscita converge in modo monotono: $\lim_{k\to\infty} |y^*(k+1) - y(k+1)|$
- Il sistema a ciclo chiuso è BIBO stabile – $y(k), u(k)$ limitati.

7 Applicazione e analisi sul pendolo

In questa sezione vengono mostrati brevemente gli algoritmi e gli schemi di controllo CFDL-MFAC e PFDL-MFAC applicati al modello del pendolo mostrato in figura e ne analizzeremo il comportamento. Lo scopo è quello di manipolare il pendolo e controllarlo con le tecniche DDC appena citate, regolando l'angolo ad un valore costante $\frac{\pi}{6}$.

Modello del pendolo

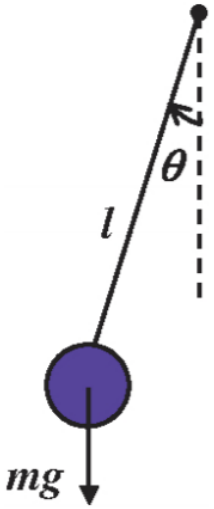

Figura n.3: Modello del pendolo

$$J\ddot{\theta}(t) = -Mglsin(\theta) - b\dot{\theta} + u(t)$$

La variabile θ è l'angolo descritto in figura ed è l'uscita da controllare, u(t) è il segnale di controllo ovvero la coppia fornita dal motore che muove il giunto.

Il parametro J è il momento di inerzia del pendolo, m la massa, l la lunghezza, g l'accelerazione di gravità, b il coefficiente di attrito. I valori dei parametri sono i seguenti:

$$M = 2.24\ kg;\ l = 0.3\ m;\ J = 0.058\ kg \cdot m^2;$$

$$g = 9.81\ {}^{m}/_{s^2};\ b = 0.0014\ kg \cdot {}^{m^2}/_{s}.$$

7.1 Simulazione pendolo CFDL-MFAC

Per prima cosa, nello schema Simulink è stato utilizzato un blocco Matlab Function per poter realizzare il controllore che sarà lo stesso per tutte le altre applicazioni del CFDL-MFAC che vedremo in seguito. L'algoritmo di controllo del CFDL-MFAC è il seguente:

```
function [phi_i,u_i,DeltaU_usc,Deltay,yPPD_i1]= fcn( e, y_i, y_i1,yPPD_i, phi_i_11,u_i1,DeltaU1,epsistima, ro, mistima, etastima, lambda, phi11)

    Deltay=y_i1-y_i;

      phi_i_1=phi_i_11;

      DeltaU=DeltaU1;

      phi_i=phi_i_1+ (etastima*DeltaU*(Deltay-phi_i_1'*DeltaU)/(mistima+norm(DeltaU)^2));


    if (norm(phi_i)<=epsistima) || norm(DeltaU)<=epsistima || (sign(phi_i(:,1))) ~= sign(phi11)
       phi_i(:,1)=phi11;
    end

    DeltayPPD=phi_i'*DeltaU;

    yPPD_i1 = yPPD_i+DeltayPPD;

    u_i=u_i1+(ro*phi_i(:,1)*e/(lambda+phi_i(:,1)^2));

  DeltaU_1=u_i-u_i1;
   DeltaU_usc=DeltaU_1;
```

Figura n.4: Algoritmo di controllo CFDL

Nello stesso schema vengono anche utilizzati dei blocchi Unit Delay Z^{-1} per ottenere i precedenti passi di campionamento. Il processo è stato anch'esso realizzato mediante un blocco Matlab Function – che a differenza del controllore, varierà per le applicazioni che vedremo successivamente. Il processo del pendolo è descritto dalla seguente funzione:

```
function y_i1 = fcn(u_i,y_i,y_i_1,M, J, b, g, l)


y_i1= ((-M*g*l*sin(y_i_1)-b*y_i+u_i)/J);
```

Figura n.5: Processo del pendolo

Inoltre, vengono anche utilizzati dei blocchi Integrator che ci consentono di integrare a tempo continuo il segnale che ricevono in ingresso. Lo schema Simulink totale è il seguente:

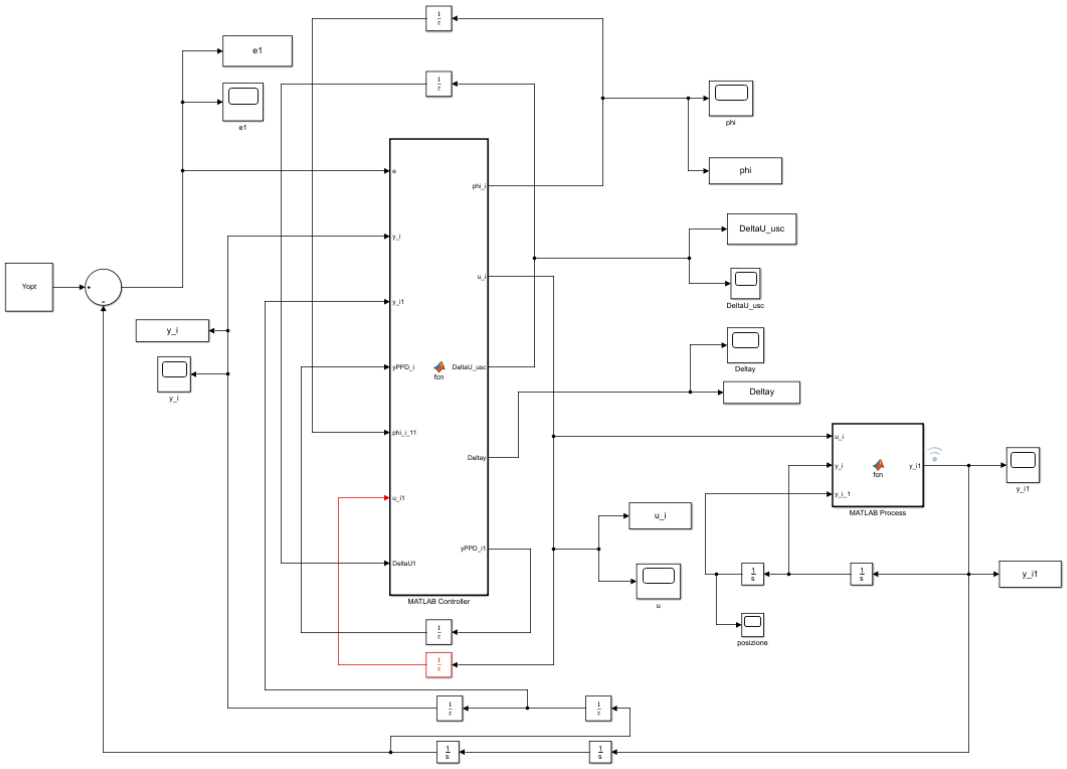

Figura n.6: Schema del pendolo CFDL

Notiamo che l'uscita del nostro processo è l'accelerazione angolare e che per ottenere la posizione è necessario integrarla due volte.

A seguito di numerose prove, sono stati individuati i seguenti valori dei parametri che si trovano all'interno dell'algoritmo di controllo CFDL-MFAC:

```
ro = 1;
epsistima = 10^-5;
etastima = 0.1;
phi11 = 0.8;
mistima = 0.01;
lambda = 1.5;
M = 2.24;
g = 9.81;
l = 0.3;
b = 0.0014;
J = 0.058;

Tc = 0.1;
Yopt = pi/6;
```

Figura n.7: Parametri algoritmo CFDL per il pendolo

Con lo scopo di analizzare al meglio il comportamento del pendolo controllato con il CFDL-MFAC, è stato creato uno schema Simulink che ci permette di confrontarlo con il comportamento del pendolo controllato con un semplice PID:

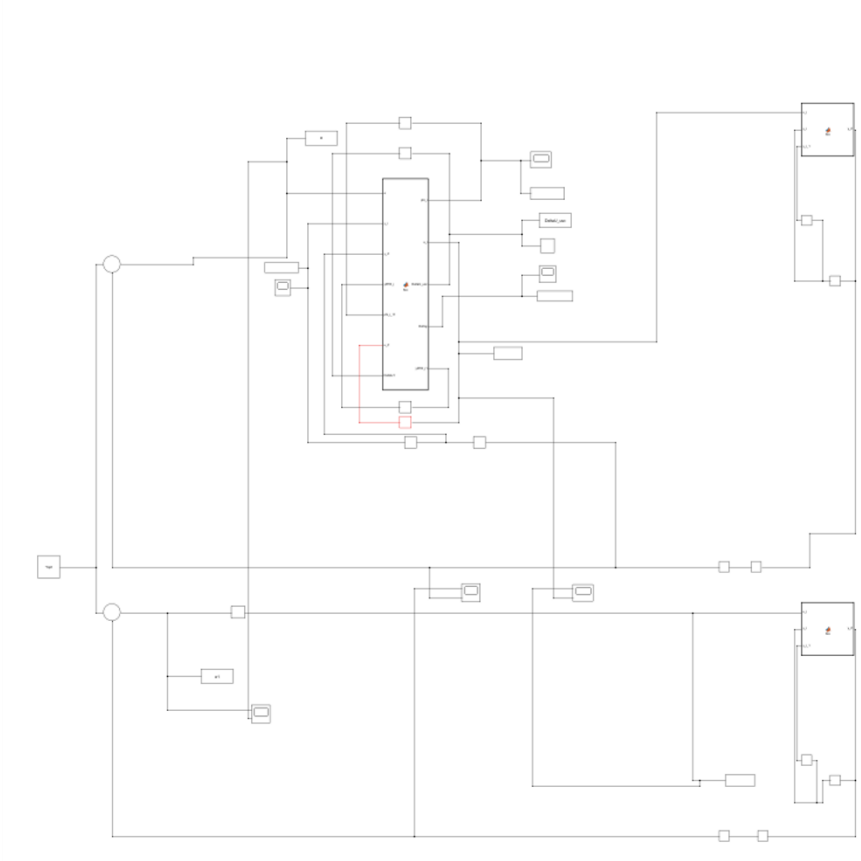

Figura n.8: Schema del pendolo CFDL-PID

Dove il PID è stato settato con i seguenti valori:

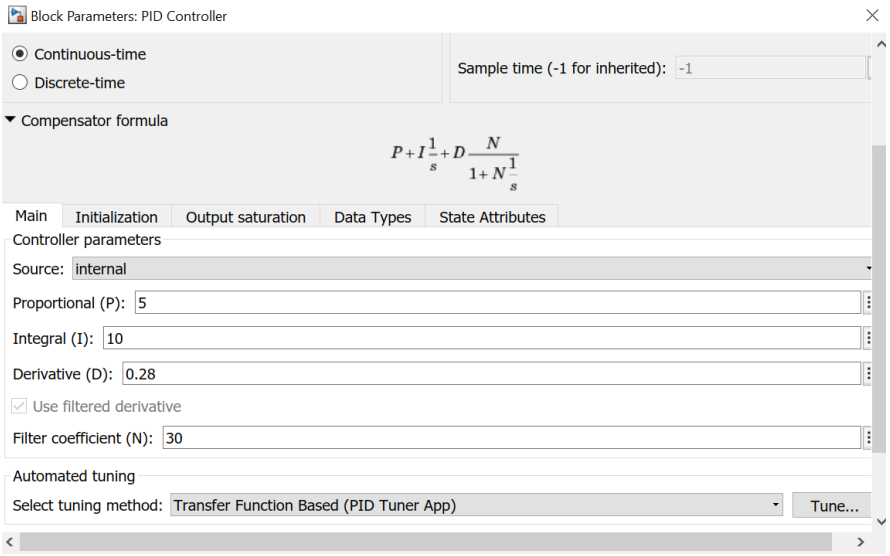

Figura n.10: Set del PID

Sotto sono mostrati i grafici dell'errore, dello sforzo di controllo e della posizione. Ricordiamo che l'errore dovrà tendere a zero, lo sforzo di controllo dovrà diventare costante con un valore non troppo elevato, mentre la posizione dovrà tendere al valore di riferimento (che in questo caso è $\frac{\pi}{6}$).

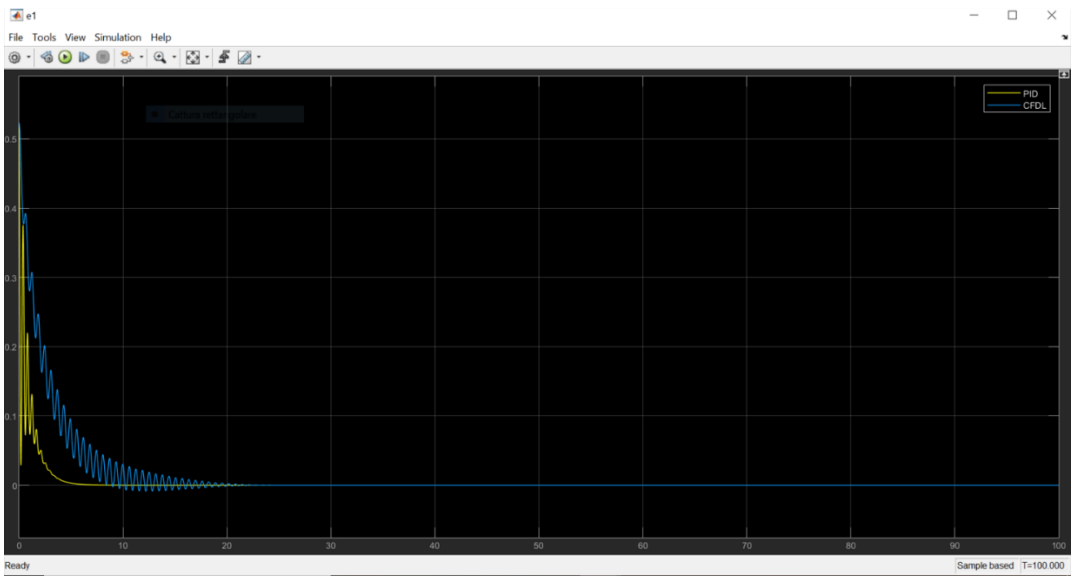

Figura n.11: Errore del pendolo con il PID (giallo) ed errore del pendolo con CFDL (blu)

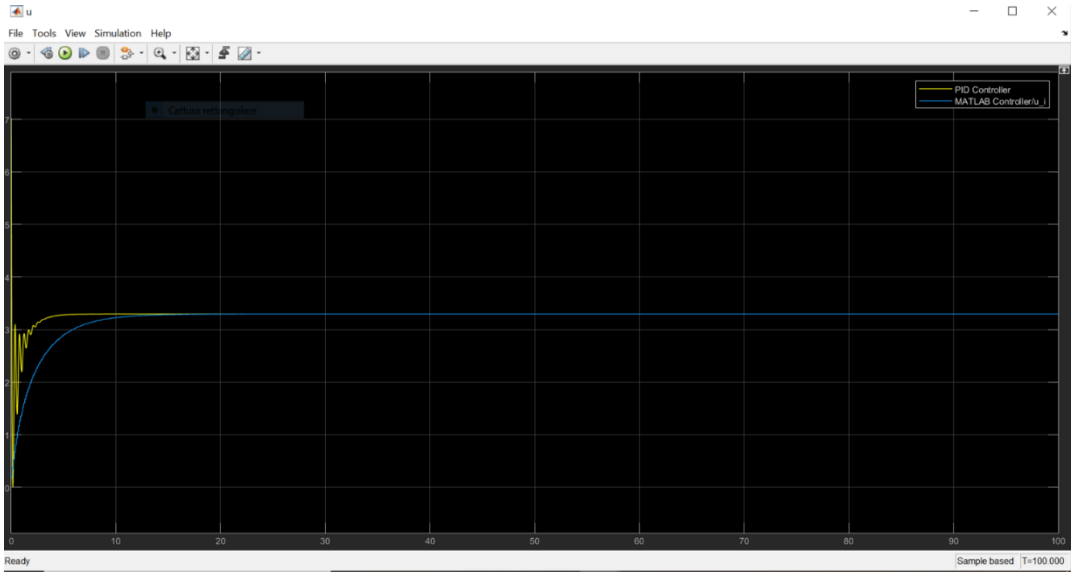

Figura n.12: Sforzo di controllo del pendolo con il PID (giallo) e sforzo di controllo del pendolo con CFDL (blu)

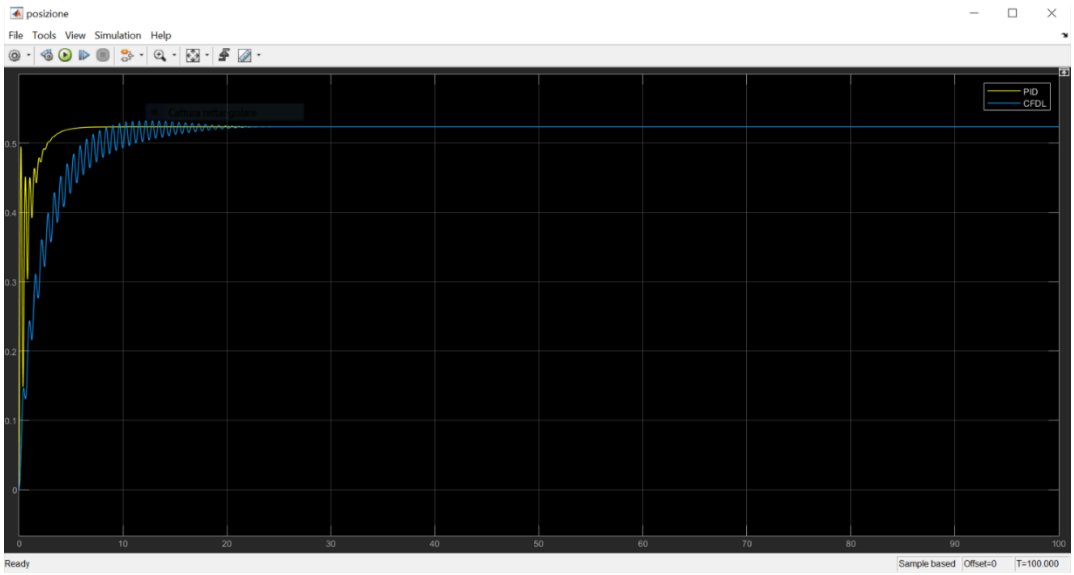

Figura n.13: Posizione del pendolo con il PID (giallo) e posizione del pendolo con CFDL (blu)

I grafici in giallo rappresentano il comportamento ottenuto con il PID, mentre i grafici in blu rappresentano il comportamento ottenuto con il CFDL-MFAC. Ci accorgiamo subito che il PID è nettamente più veloce rispetto al CFDL-MFAC, ma ha un andamento più oscillatorio all'inizio.

In generale notiamo che con il CFDL-MFAC all'incirca dopo 30 secondi otteniamo il comportamento desiderato e quindi il sistema converge e si stabilizza. Questo risultato è conseguenza della combinazione dei parametri scelti.

7.2 Simulazione pendolo PFDL-MFAC

Anche qui, nello schema Simulink è stato utilizzato un blocco Matlab Function per poter realizzare il controllore che sarà lo stesso per tutte le altre applicazioni del PFDL-MFAC che vedremo in seguito. L'algoritmo di controllo del PFDL-MFAC è il seguente:

```
function [phi_i,u_i,DeltaU_usc,Deltay,yPPD_i1]= fcn(e,y_i,y_i1,yPPD_i,phi_i_11,phi_i_12,u_i1,u_i2,DeltaU1,DeltaU2,epsistima,ro,mistima,etastima

    Deltay=y_i1-y_i;

    phi_i_1=[phi_i_11,phi_i_12];

    DeltaU=[DeltaU1,DeltaU2];

    phi_i=phi_i_1+ (etastima*DeltaU*(Deltay-phi_i_1'*DeltaU)/(mistima+norm(DeltaU)^2));

  if (norm(phi_i)<=epsistima) || norm(DeltaU)<=epsistima || (sign(phi_i(:,1))) ~= sign(phi11)
     phi_i(:,1)=phi11;
     phi_i(:,2)=phi12;
  end

  DeltayPPD=phi_i'*DeltaU;
  yPPD_i1 = yPPD_i+DeltayPPD;

  u_i=u_i1+(ro*phi_i(:,1)*e/(lambda+phi_i(:,1)^2))-phi_i(:,1)*((ro1*phi_i(:,2)*DeltaU(2))/(lambda+phi_i(:,1)^2));

DeltaU_1=u_i-u_i1;
DeltaU_2=u_i1-u_i2;
DeltaU_usc=[DeltaU_1,DeltaU_2];
```

Figura n.14: Algoritmo di controllo del PFDL

Il processo del pendolo è lo stesso visto nella precedente Figura n.5. Anche qui vengono utilizzati i blocchi Unit Delay Z^{-1} e Integrator per gli stessi motivi di cui sopra. Lo schema Simulink totale è il seguente:

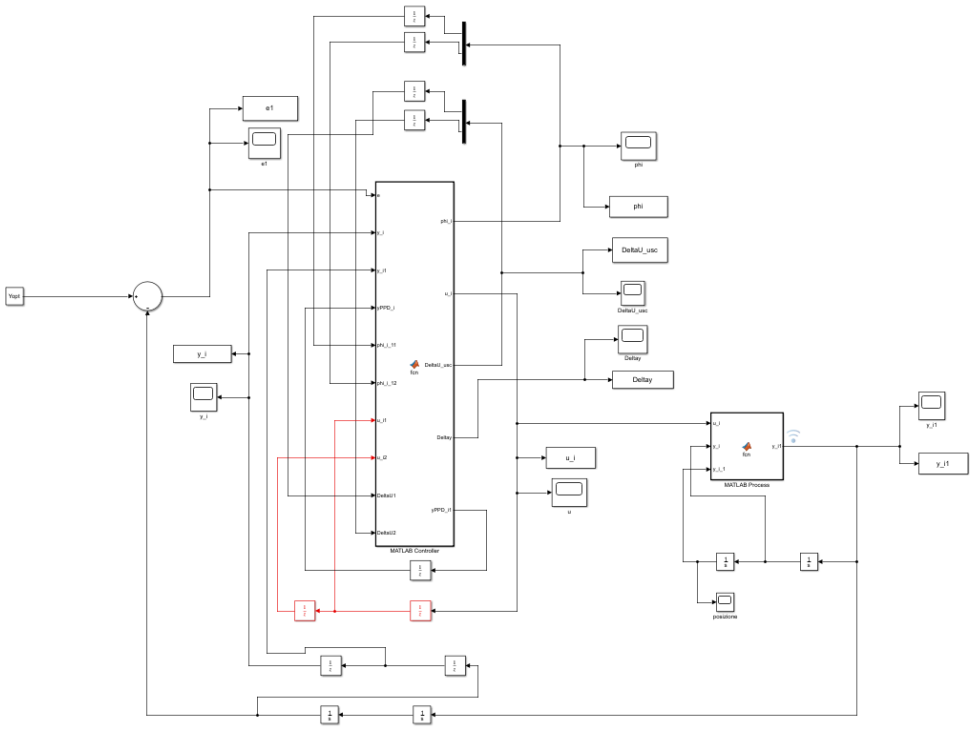

Figura n.15: Schema del pendolo PFDL

Anche qui l'uscita del nostro processo è l'accelerazione angolare e per ottenere la posizione è necessario integrarla due volte.

Dopo numerose prove, sono stati individuati i seguenti valori dei parametri che si trovano all'interno dell'algoritmo di controllo PFDL-MFAC:

```
ro = 0.99;
ro1= 0.99;
epsistima = 10^-5;
etastima = 0.01;
phi11 = 0.99;
phi12 = 0;
mistima = 0.1;
lambda = 3;
M= 2.24;
l = 0.3;
J = 0.058;
g = 9.81;
b = 0.0014;
Yopt = pi/6;
Tc = 0.1;
```

Figura n.16: Parametri algoritmo PFDL per il pendolo

Con lo scopo di analizzare al meglio il comportamento del pendolo controllato con il PFDL-MFAC, è stato creato uno schema Simulink che ci permette di confrontarlo con il comportamento del pendolo controllato con un semplice PID:

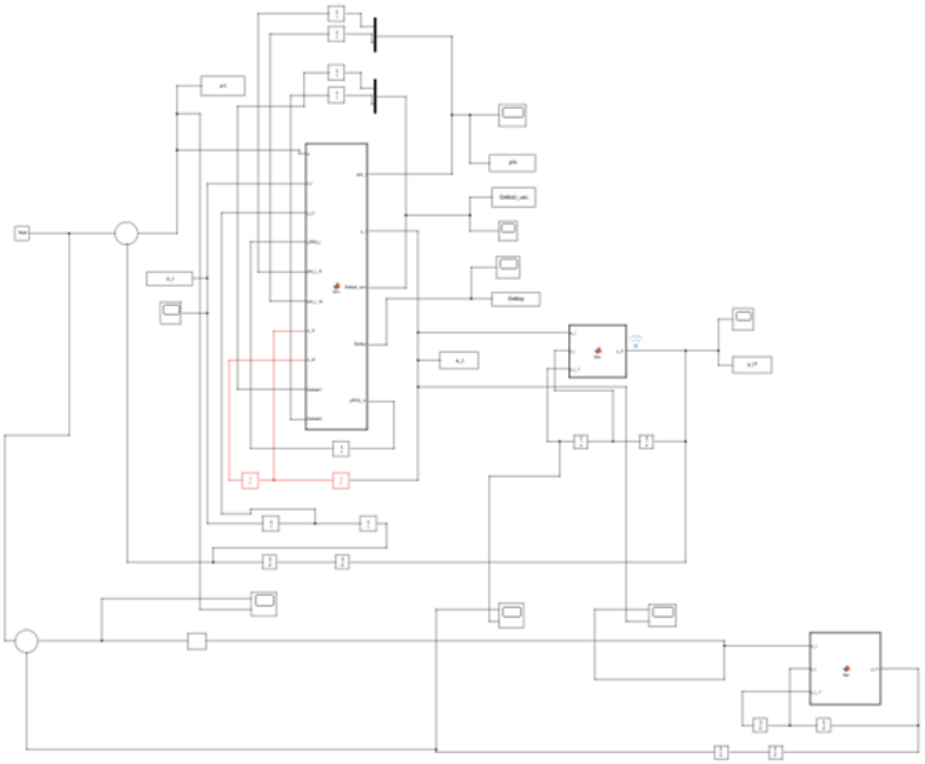

Figura n.17: Schema del pendolo PFDL-PID

Dove il PID è stato settato con gli stessi valori mostrati in Figura n.10.

Sotto sono mostrati i grafici dell'errore, dello sforzo di controllo e della posizione. Ricordiamo che l'errore dovrà tendere a zero, lo sforzo di controllo dovrà diventare costante con un valore non troppo elevato, mentre la posizione dovrà tendere al valore di riferimento (che in questo caso è $\frac{\pi}{6}$).

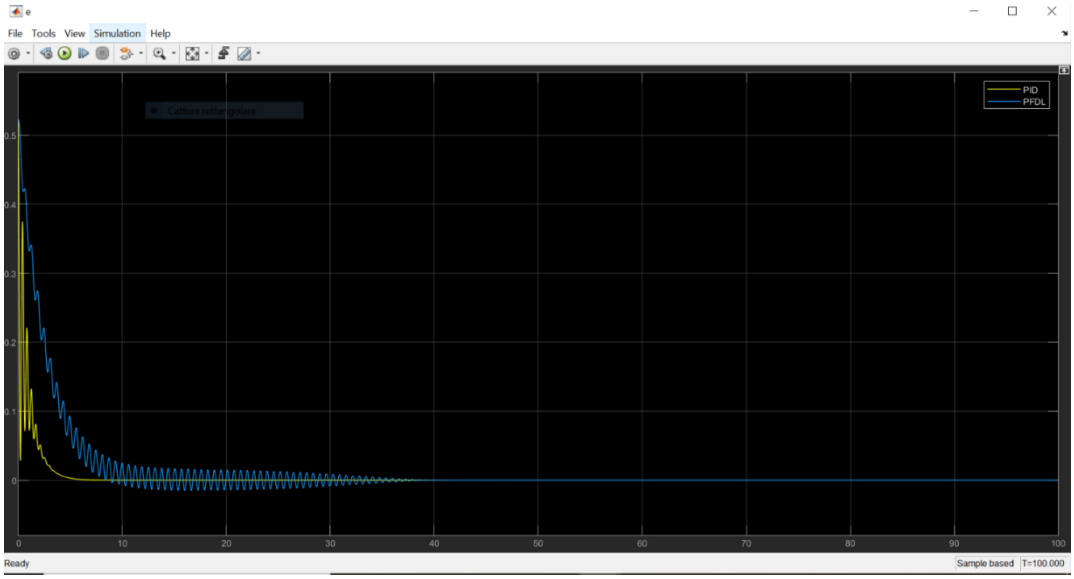

Figura n.18: Errore del pendolo con il PID (giallo) ed errore del pendolo con PFDL (blu)

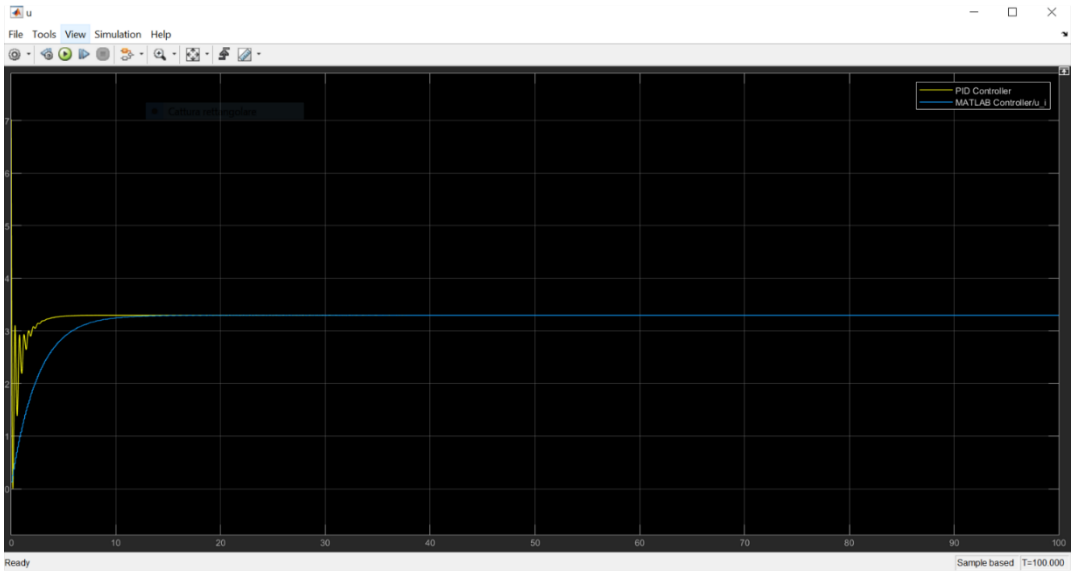

Figura n.19: Sforzo di controllo del pendolo con il PID (giallo) e sforzo di controllo del pendolo con PFDL (blu)

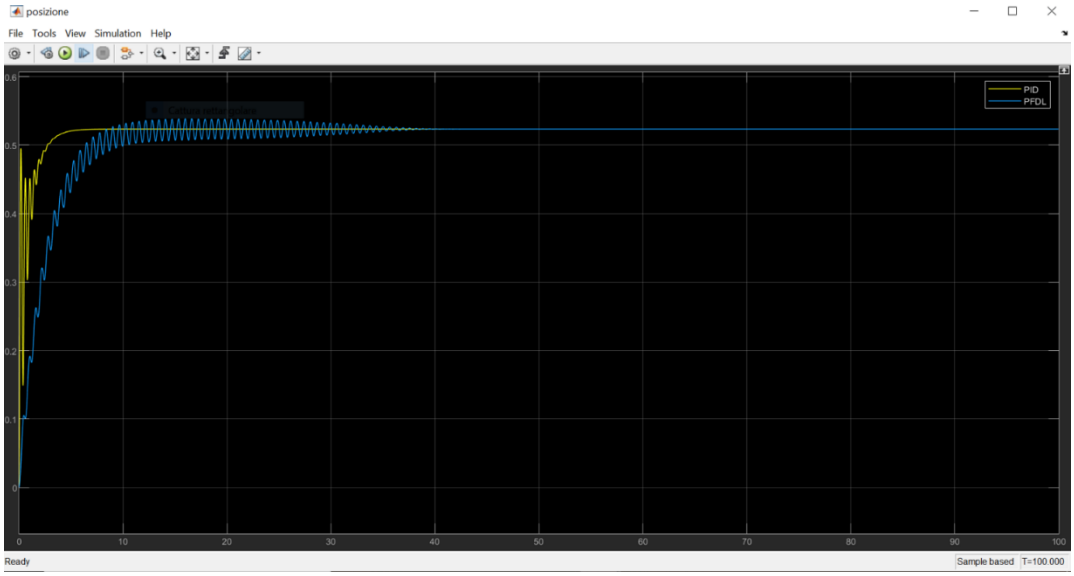

Figura n.20: Posizione del pendolo con il PID (giallo) e posizione del pendolo con PFDL (blu)

I grafici in giallo rappresentano il comportamento ottenuto con il PID, mentre i grafici in blu rappresentano il comportamento ottenuto con il PFDL-MFAC. Ci accorgiamo che anche qui il PID è molto più veloce rispetto al PFDL-MFAC, ma ha un andamento più oscillatorio all'inizio. In generale notiamo che con il PFDL-MFAC all'incirca dopo 40 secondi otteniamo il comportamento desiderato e quindi il sistema converge e si stabilizza; anche se effettuando lo zoom sui grafici è possibile notare come il tempo di salita in questo caso sia minore rispetto a quello riscontrato nel CFDL-MFAC.

8 Quadrirotori

8.1 Generalità

Un quadrirotore è un aeromobile innalzato e spinto da quattro rotori. I quadrirotori vengono classificati come aerogiri, diversi dagli aeroplani in quanto la portanza è data esclusivamente dai quattro rotori. Essi sono impropriamente classificati anche come elicotteri, sebbene i quadrirotori hanno la funzione di utilizzare pale a "passo fisso", ovvero aventi attacco rigido con il mozzo, il cui angolo d'attacco non varia durante la rotazione.

I quadrirotori di ultima generazione sono stati progettati per poter volare anche in assenza di pilota umano a bordo, ovvero col fine di ottenere un APR (Aeromobile a Pilotaggio Remoto). In particolare, questi velivoli utilizzano sensori elettrici e dei sistemi di controllo per stabilizzarsi.

La grande stabilità e la facilità di controllo dei quadrirotori è dovuta alla configurazione a croce dei due assi sui quali sono disposti i quattro motori. Inoltre, notiamo che esso è un sistema sotto-attuato, in quanto possiede sei gradi di libertà:

1. Beccheggio (oscillazione del velivolo attorno ad un asse trasversale);
2. Imbardata (oscillazione del velivolo attorno all'asse verticale passante per il baricentro);
3. Rollio (oscillazione del velivolo attorno al proprio asse longitudinale);
4. x (movimento nella direzione frontale del veicolo);
5. y (movimento verso il lato sinistro del veicolo);
6. z (altitudine);

ma viene controllato utilizzando solo quattro attuatori. In particolare, il controllo di un quadricottero consiste nel regolare le velocità angolari dei rotori che vengono azionate dai motori elettrici.

Come possiamo notare nella Figura n.21, due rotori opposti girano nello stesso verso, mentre i rotori adiacenti girano nel verso opposto. Nel particolare caso in cui tutti e quattro i rotori ruotano con la stessa velocità angolare, con i rotori 1 e 3 che girano in senso destrogiro, mentre i rotori 2 e 4 girano in senso levogiro, otteniamo accelerazione angolare nulla in corrispondenza dell'asse d'imbardata; ne segue che il rotore posteriore generalmente utilizzato negli elicotteri tradizionali non è necessario.

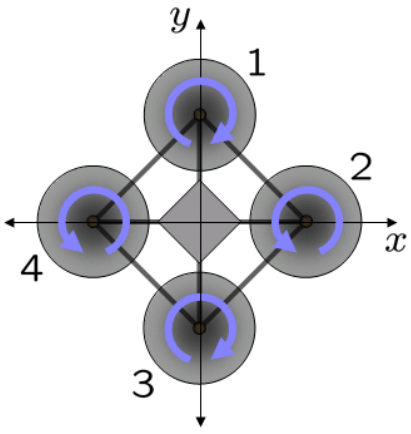

Figura n.21: Schema della movimentazione quadricottero

L'imbardata (generazione dell'angolo Yaw), invece, può essere ottenuta generando una certa differenza tra le velocità dei rotori che ruotano in senso destrogiro e di quelli che girano in senso levogiro, mantenendo comunque la stessa velocità angolare per i rotori che si trovano sullo stesso asse.

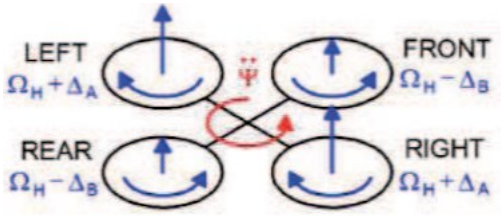

Figura n.22: Imbardata

La salita è lo spostamento più semplice e lo si può ottenere fornendo la stessa velocità angolare a tutti e quattro i rotori, generando così quattro forze di spinta identiche.

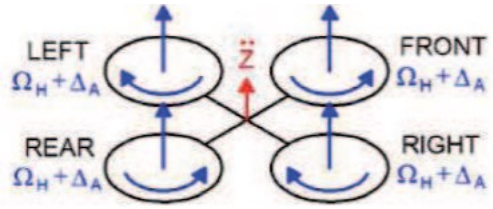

Figura n.23: Salita

Le manovre di rollio e beccheggio possono essere effettuate senza influenzare l'asse d'imbardata. Queste due manovre hanno sostanzialmente lo stesso funzionamento. Il rollio (generazione dell'angolo Roll) consiste nel far ruotare alla stessa velocità angolare i rotori che si trovano lungo l'asse x, mentre gli altri due rotori vengono fatti ruotare con velocità angolari distinte tra loro. Il beccheggio (generazione dell'angolo Pitch) consiste nel far ruotare alla stessa velocità angolare i rotori che si trovano lungo l'asse y, mentre gli altri due rotori vengono fatti ruotare con velocità angolari distinte tra loro.

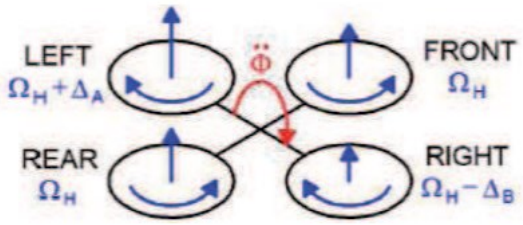

Figura n.24: Rollio

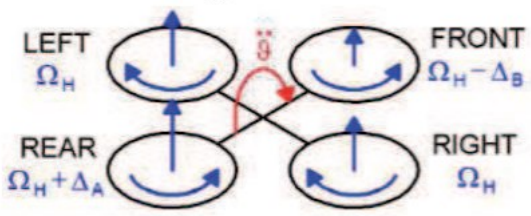

Figura n.25: Beccheggio

8.2 Modellazione Matematica

In questa sezione viene presentata l'accurata modellazione matematica del drone, ma pur sempre tenendo conto di alcune semplificazioni riguardo la sua struttura meccanica in modo tale da rendere il più chiaro possibile la logica di funzionamento del velivolo.

La modellazione matematica generalmente utilizzata è quella che si basa sul considerare la dinamica di un corpo rigido sottoposto a forze e coppie. Ovvero, si considerano:

- Struttura perfettamente simmetrica, costituita da un telaio rigido e quattro motori che vengono supposti identici e quindi l'analisi del solo motore i-esimo non perde di generalità
- Velivolo tempo-invariante, che implica massa invariante durante il moto
- Centro di gravità fissato nell'origine
- Costanti di spinta e di resistenza legate alla velocità del motore al quadrato

Generalmente, s'ipotizza che il blade flapping sia assente, ma nelle equazioni a seguire ne terremo conto. Il blade flapping è quel fenomeno consiste nel movimento "su e giù" della pala del rotore che provoca l'eliminazione della dissimmetria della portanza. La pala mentre si innalza, incontra velocità progressivamente più elevate, rispondendo all'aumento della velocità producendo, di conseguenza, più portanza. La pala si alza dunque verso l'alto e la variazione del vento relativo e dell'angolo di attacco riduce la quantità che sarebbe stata generata.

Vi sono tre diversi sistemi di riferimento su cui si farà riferimento e su cui si baserà l'intero sviluppo dei sei gradi di libertà del quadrirotore – di cui tre movimenti sono traslatori e tre sono rotatori.

1. Sistema di riferimento inerziale I = {Ex, Ey, Ez}, usato facendo riferimento al piano tangente con la superficie della Terra prendendo come origine la posizione dell'osservatore.
2. Sistema di riferimento solidale con il drone o body frame A = {E1, E2, E3}, centrato nel centro di massa (CdM) del velivolo.
3. Sistema di riferimento allineato con il quadro di riferimento inerziale ξ = (x, y, z), centrato nel centro di gravità del velivolo.

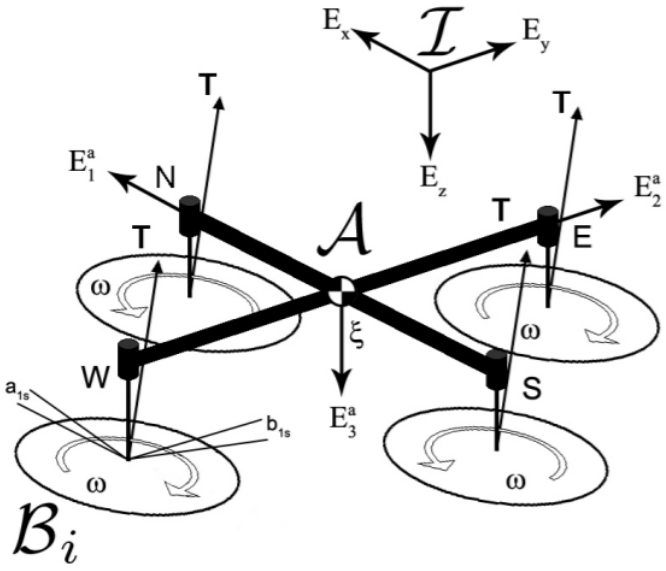

Figura n.26: Sistemi di riferimento usati nel drone

Per poter mettere in relazione i primi due sistemi di riferimento visti si utilizza una matrice di rotazione R: A→I. Essa ci consente di ottenere lo spostamento angolare rispetto al sistema di riferimento inerziale; a differenza dello spostamento lineare che, una volta nota la posizione del velivolo nello spazio, può essere direttamente ricavata dal sistema di riferimento inerziale I.

Gli angoli di Eulero (ϕ, θ, ψ) possono essere utilizzati parametrizzare la rotazione di un corpo rigido nello spazio e in tal caso rappresentano matematicamente le tre rotazioni sugli assi di riferimento. Tenendo conto che la nostra terna (x,y,z) rispetta la regola della mano destra -con x positiva se diretta verso il fronte del drone, y positiva se diretta verso il lato destro del drone e z positiva se diretta verso il basso – possiamo descrivere le tre seguenti matrici di rotazione:

- R(x, ϕ): rotazione attorno all'asse x
- R(y, θ): rotazione attorno all'asse y,
- R(z, ψ): rotazione attorno all'asse z.

Dove:

$$R(x,\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi & cos\phi \end{bmatrix}; \quad (19)$$

$$R(y,\theta) = \begin{bmatrix} cos\theta & 0 & sen\theta \\ 0 & 1 & 0 \\ -sen\theta & 0 & cos\phi \end{bmatrix}; \quad (20)$$

$$R(z,\psi) = \begin{bmatrix} cos\psi & -sin\psi & 0 \\ sin\psi & cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad (21)$$

D'ora in poi indicheremo cos con l'abbreviazione c e sen con l'abbreviazione s per non appesantire troppo la scrittura delle formule. La matrice di rotazione complessiva è data da:

$$R(\phi,\theta,\psi) = R(x,\phi)\,R(y,\theta)R(z,\psi); \quad (22)$$

$$\boldsymbol{R_b^i} = \begin{bmatrix} c\psi c\theta & c\psi s\theta s\phi - s\psi c\phi & c\psi s\theta c\phi + s\psi s\phi \\ s\phi c\theta & s\psi s\theta s\phi + c\psi c\phi & s\psi s\theta c\phi - s\phi c\psi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix}; \quad (23)$$

La matrice (23) ha il seguente utilizzo: essa viene moltiplicata per delle coordinate espresse rispetto al sistema di riferimento A in modo tale da poterle esprimere rispetto al sistema di riferimento I. Tale matrice è ortonormale e quindi la sua inversa coincide con la sua trasposta (ovvero, $R^T = R^{-1}$).

$$\boldsymbol{R_i^b} = \begin{bmatrix} c\psi c\theta & s\psi c\theta & -s\theta \\ c\psi s\theta s\phi - s\psi c\phi & s\psi s\theta s\phi + c\psi c\phi & c\theta s\phi \\ c\psi s\theta c\phi + s\psi s\phi & s\psi s\theta c\phi - s\phi c\psi & c\theta c\phi \end{bmatrix}; \quad (24)$$

La matrice (24) è l'inversa della matrice (23) e anche il suo utilizzo è inverso: esse viene moltiplicata per delle coordinate espresse rispetto al sistema di riferimento I in modo tale da poterle esprimere rispetto al sistema di riferimento A.

Possiamo utilizzare le matrici (19), (20), (21) col fine di esprimere le velocità angolari rispetto al sistema di riferimento I in velocità angolari rispetto al sistema di riferimento A. Allo stesso tempo, vengono introdotti i vettori velocità di traslazione (v) e velocità di rotazione (ω^b), dove:

$$\omega^b = R(\phi)R(\theta)R(\psi)\begin{bmatrix} \dot{\psi} \\ 0 \\ 0 \end{bmatrix} + R(\phi)R(\theta)\begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + R(\phi)\begin{bmatrix} 0 \\ 0 \\ \dot{\phi} \end{bmatrix}; \quad (25)$$

Dalla quale otteniamo:

$$\omega^b = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s\theta \\ 0 & c\phi & c\theta s\phi \\ 0 & -s\phi & c\theta c\phi \end{bmatrix}\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix}; \quad (26)$$

L'inversa della matrice (26) può essere utilizzata col fine di esprimere le velocità angolari rispetto al sistema di riferimento A in velocità angolari rispetto al sistema di riferimento I. Effettuando tale passaggio otteniamo:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix} = \begin{bmatrix}1 & s\phi\, {s\theta}/{c\theta} & c\phi\, {s\theta}/{c\theta}\\ 0 & c\phi & -s\phi\\ 0 & s\phi\, {1}/{c\theta} & c\theta\, {1}/{c\theta}\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}; \quad (27)$$

Dalla (27) è possibile affermare che si può avere imbardata in seguito a manovre di rollio o beccheggio, infatti possiamo notare come la variazione dell'angolo ψ dipenda da quelle di ϕ e θ.

8.3 Equazioni della dinamica

Per comprendere al meglio le equazioni a seguire si faccia riferimento alla tabella dedicata ai simboli in fondo alla tesi. Le equazioni scritte sotto rappresentano le formule per ricavare le velocità lineari, le velocità angolari, le loro rispettive accelerazioni, forze, coppie e momenti dipendenti dalle spinte prodotte dai motori.

$$\dot{\xi} = \boldsymbol{R}v; \quad (28)$$

L'equazione (28) individua la velocità del CdM espressa in termini di sistema di riferimento inerziale, servendosi della matrice di trasformazione R.

$$m\dot{\boldsymbol{v}} = -m\omega \times \boldsymbol{v} + mg\boldsymbol{R^T}e_3 + \sum_{N,S,E,W} \boldsymbol{t}_i; \quad (29)$$

L'equazione (29) rappresenta la seconda Legge di Newton applicata ad un corpo rigido – in questo caso il drone: l'accelerazione del corpo è data dalla somma di tutte le forze che agiscono su di esso diviso la massa. Come possiamo intuire dalla formula, le forze che agiscono sul drone sono: la forza di attrito viscoso, la forza di gravità e le quattro forze di spinta generate dai quattro motori.

Il momento meccanico totale $I\dot{\omega}$ è dato dal prodotto vettoriale tra il momento angolare e la velocità sommato al contributo meccanico fornito da ciascun singolo motore. Otteniamo dunque che l'equazione (30) è equivalente alla (29), in termini di dinamica rotazionale.

$$I\dot{\omega} = -\omega \times I\omega + \sum_{N,S,E,W}[\boldsymbol{q}_i + \boldsymbol{m}_i]; \quad (30)$$

La forza di spinta dell'i-esimo motore è determinata dalla seguente equazione:

$$\boldsymbol{t}_i = C_T\rho A r^2 \omega_i^2 \begin{pmatrix} -sina_{1i} \\ cosa_{1i}sinb_{1i} \\ -cosb_{1i}cosa_{1i} \end{pmatrix}; \quad (31)$$

Dall'equazione (31) possiamo notare che la forza di spinta è direttamente proporzionale ai parametri inerenti alla struttura dell'elica (area e raggio), alla densità del fluido in cui è immerso e al coefficiente di spinta. Particolare importanza viene posta alla velocità di rotazione elevata al quadrato, in quanto costituisce un grande contributo per l'equazione. La moltiplicazione per il vettore a destra viene fatta per tenere conto dell'effetto blade flapping, dove a_{1i} e b_{1i} sono gli angoli

formati dagli assi longitudinali e laterali del velivolo rispetto alla direzione di avanzamento. La direzione di spinta è sempre ortogonale al piano su cui giace il drone.

La coppia prodotta dall'i-esimo motore, ovvero il contributo fornito dal momento di drag, è determinata dalla seguente equazione:

$$\boldsymbol{q}_i = C_Q \rho A r^3 \omega_i |\omega_i| e_3; \quad (32)$$

Anche nell'equazione (32) vi sono i parametri inerenti alla struttura dell'elica (area e raggio) e la densità del fluido in cui è immerso. Notiamo però che la coppia è diretta solo lungo l'asse verticale (z) a causa della moltiplicazione per e_3. Viene inoltre determinato il valore assoluto della velocità angolare moltiplicato per sé stesso in modo tale da preservare il segno che rappresenterà il verso di rotazione. La (32) sostanzialmente rappresenta il momento generato dalle forze resistenti dell'aerodinamica, le quali agiscono sulle eliche e vengono trasmesse al rotore, agendo su di esso in relazione al suo raggio.

Il momento di spinta dell'i-esimo motore è determinato dalla seguente equazione:

$$\boldsymbol{m}_i = \boldsymbol{t}_i \times \boldsymbol{d}_i; \quad (33)$$

Notiamo che la seconda parte dell'equazione (33) è il prodotto vettoriale tra la forza di spinta e la distanza tra i rotori determinata tramite le matrici (19), (20), (21).

Le espressioni (34) e (35) ci consentono di effettuare il passaggio dagli angoli u_{1i} e v_{1i} agli angoli a_{1i} e b_{1i}:

$$\boldsymbol{J_B} = \begin{pmatrix} cos\psi_{ri} & -sin\psi_{ri} \\ sin\psi_{ri} & cos\psi_{ri} \end{pmatrix}; \quad (34)$$

$$\begin{pmatrix} a_{1i} \\ b_{1i} \end{pmatrix} = \boldsymbol{J_B} \begin{pmatrix} u_{1i} \\ v_{1i} \end{pmatrix}; \quad (35)$$

La matrice J_B ci consente di trasformare le coordinate rispetto al sistema di riferimento solidale col corpo del drone con le coordinate tra il piano (Ex; Ey) del sistema di riferimento inerziale.

I vettori distanza - ce n'è uno per ogni motore - ci consentono di individuare il posizionamento dei rotori nello spazio tramite (36), (37), (38), (39), dove la sigla (NSEW) corrisponde all'orientamento rispetto al nord magnetico terrestre (North, South, East, West).

$$\boldsymbol{d_N} = (0 \quad d \quad h); \quad (36)$$

$$\boldsymbol{d_S} = (0 \quad -d \quad h); \quad (37)$$

$$\boldsymbol{d_E} = (d \quad 0 \quad h); \quad (38)$$

$$\boldsymbol{d_W} = (-d \quad 0 \quad h); \quad (39)$$

Compiendo movimenti di pitch e roll, i rotori producono una velocità verticale e di conseguenza anche una spinta verticale. Quest'ultime possono essere messe in relazione tramite la formula (40):

$$\frac{C_T}{\sigma} = \frac{a(\alpha)}{4}\left[\theta_{tip} - \frac{v_i - V_c}{\omega r}\right]; \quad (40)$$

Nelle formule viste finora C_T e C_Q sono state trattate come costanti.

Le espressioni che vedremo qui sotto, invece, riguardano l'effetto blade flapping. Il vettore velocità del singolo motore è determinato dalla seguente equazione (41):

$$\boldsymbol{v}_{ri} = \boldsymbol{v} + \omega \times \boldsymbol{d}_i; \quad (41)$$

Notiamo come essa dipenda dal prodotto vettoriale tra la velocità angolare e il vettore distanza dell'i-esimo motore.

L'advance ratio del singolo motore è determinata dalla seguente equazione (42):

$$\mu_{ri} = \frac{||\boldsymbol{v}_{r(1,2)i}||}{\omega_i r}; \quad (42)$$

Il numeratore di (42) rappresenta il modulo del vettore velocità lineare calcolato lungo i primi due assi: quello longitudinale e quello laterale; mentre il denominatore è la velocità angolare dello stesso motore considerato.

L'angolo che viene generato tra la direzione di avanzamento e il nord terrestre è determinato dalla seguente equazione (43):

$$\psi_{ri} = arctg\left(\frac{\boldsymbol{v}_{r(2)i}}{\boldsymbol{v}_{r(1)i}}\right); \quad (43)$$

Gli angoli che possiamo notare in Figura n.26 sono generati dall'effetto del blade flapping e possono essere ricavati nel seguente modo:

$$u_{1_s i} = \frac{1}{1 - \frac{\mu_{ri}^2}{2}}\mu_{ri}(4\theta_t - 2\lambda_{hi}); \quad (44)$$

Ma nel nostro caso dobbiamo ricordare che λ_{hi}è l'inflow ratio ed è una costante: $\lambda_{hi} = \sqrt{C_T/2}$. Il contributo laterale è nullo ed è presente solo quello longitudinale.

Considerando le equazioni descritte sopra, esse ci consentono di calcolare le forze cui il drone è sottoposto, per poi ricavare le sei equazioni differenziali che descrivono il moto del drone e raggrupparle in un'unica grande matrice dalla quale possiamo partire per ricavare i sottosistemi di ogni grandezza.

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \\ \ddot{\phi} \\ \ddot{\theta} \\ \ddot{\psi} \end{bmatrix} = \begin{pmatrix} R_E^B \left\{ \begin{bmatrix} 0 \\ 0 \\ \sum T_i \end{bmatrix} - \begin{bmatrix} K_1\dot{x} \\ K_2\dot{y} \\ K_3\dot{z} \end{bmatrix} \right\} \frac{1}{m} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \\ (T_1 - T_2 - T_3 + T_4 + K_4\dot{\phi})\frac{1}{I_x} \\ (-T_1 - T_2 + T_3 + T_4 + K_5\dot{\theta})\frac{1}{I_y} \\ (CT_1 - CT_2 + CT_3 - CT_4 + K_6\dot{\psi})\frac{1}{I_z} \end{pmatrix}; \quad (45)$$

8.4 Funzione di Trasferimento

La dinamica lungo il punto di hover implica che lo stazionamento in volo del velivolo sia a velocità nulla. In tal caso, si viene a creare un equilibrio tra la forza peso diretta verso il basso e la forza di spinta generata dai motori verso l'alto; esse hanno stessa direzione ma verso opposto.

Ricordando che la formula per determinare una qualsiasi funzione di trasferimento dinamica è la seguente:

$$P(s) = C(sI - A)^{-1}B + D; \quad (46)$$

E tenendo contro che gli input del nostro sistema sono: dati relativi alla struttura del velivolo (es: peso, massa, …), dati acquisiti nello stato iniziale (vettori velocità lineare e angolare e vettori posizione), le forze e i momenti esterni che agiscono nello stato iniziale; possiamo concludere che analizzando il legame con cui sono connessi uscita e ingresso relativamente allo stato del sistema da controllare, si può ricavare il sottosistema desiderato dal modello lineare per quanto riguarda l'angolo di pitch.

È possibile ricavare le matrici della dinamica A, B, C, D considerando solo i coefficienti relativi alle sole variabili di stato θ e q, ottenendo:

$$A = \begin{pmatrix} 0 & 1 \\ 0 & -1.692 \end{pmatrix}, B = \begin{pmatrix} 0 \\ 0.3133 \end{pmatrix}, C = (1 \quad 0), D = 0; \quad (47)$$

Per ottenere la funzione di trasferimento finale, però, è necessario moltiplicare il sistema ottenuto inserendo le matrici (47) all'interno della funzione (46) con altri due coefficienti $K_1 = 5.6659$ e $K_2 = 1530.7$, che rispettivamente rappresentano la partizione del segnale di controllo tra i diversi motori e il coefficiente di spinta dei motori.

La funzione di trasferimento per l'angolo di pitch è la seguente:

$$P(s) = \frac{2717}{s^2 + 1.692s}; \quad (48)$$

La funzione (48) ci fornisce la dinamica semplificata del sistema, tenendo conto dei legami esistenti tra le diverse variabili che incidono su di essa.

Nel paragrafo successivo applicheremo i metodi di controllo CFDL-MFAC e PFDL-MFAC mostrati in precedenza alla funzione di trasferimento (48) e ne analizzeremo il comportamento.

9 Applicazione e analisi sul drone

In questa sezione vengono mostrati gli algoritmi e gli schemi di controllo CFDL-MFAC e PFDL-MFAC applicati alla funzione di trasferimento del sottosistema di pitch e ne analizzeremo il comportamento.

Ricordiamo che stiamo effettuando il controllo su di una funzione di trasferimento ottenuta a seguito della linearizzazione in un punto in hovering e quindi in questo caso l'angolo di pitch deve essere quasi nullo. A tal proposito, in entrambe le simulazioni sotto riportate, l'angolo è stato considerato pari a 0 e abbiamo supposto che il tempo di campionamento utilizzato dall'algoritmo sia 0.001 secondi, in quanto necessitiamo di una buona precisione ma un tempo minore sarebbe stato troppo dispendioso.

Inoltre, per entrambe le simulazioni, si è deciso di controllare la posizione (l'angolo di pitch) con un controllo a cascata, facendo uso di un PID e dell'algoritmo CFDL-MFAC o PFDL-MFAC.

9.1 Controllo in Pitch: Simulazione CFDL-MFAC

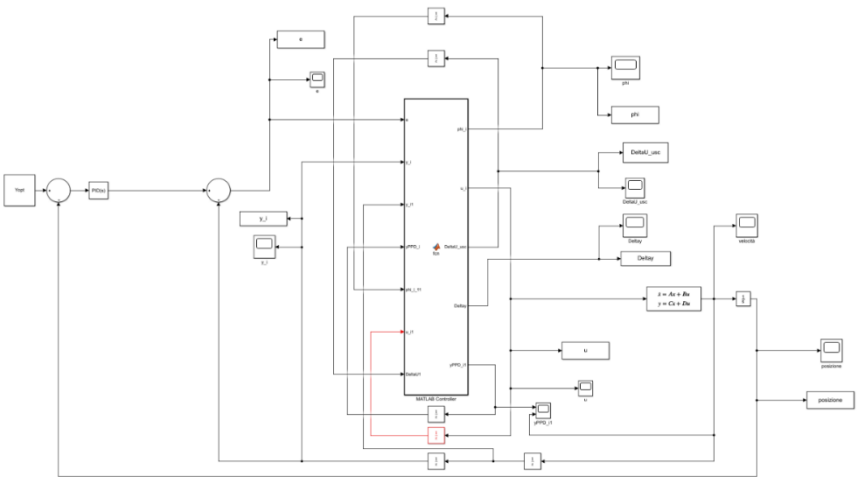

Figura n.27: Schema del drone CFDL

Anche in questo caso vengono utilizzati dei blocchi Unit Delay Z^{-1} con lo stesso scopo di cui prima.

Come possiamo notare dalla Figura n.27 sopra, il grafico della posizione lo si può ricavare integrando – con il blocco Integrator – l'uscita del blocco State-Space, ovvero la velocità.

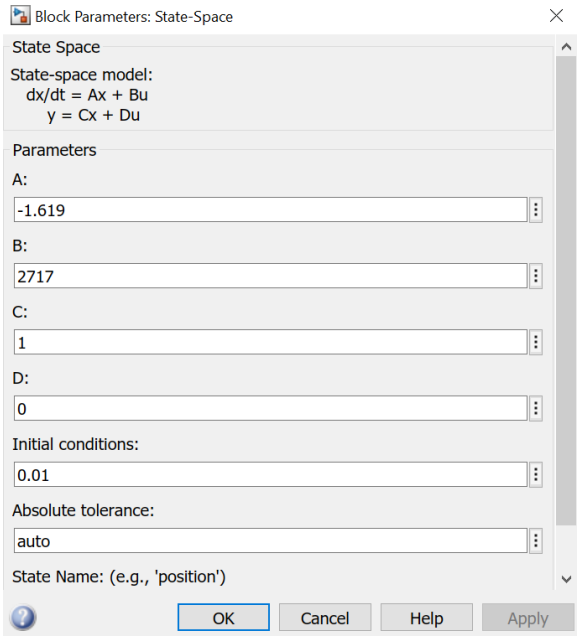

Figura n.28: Set dello State-Space

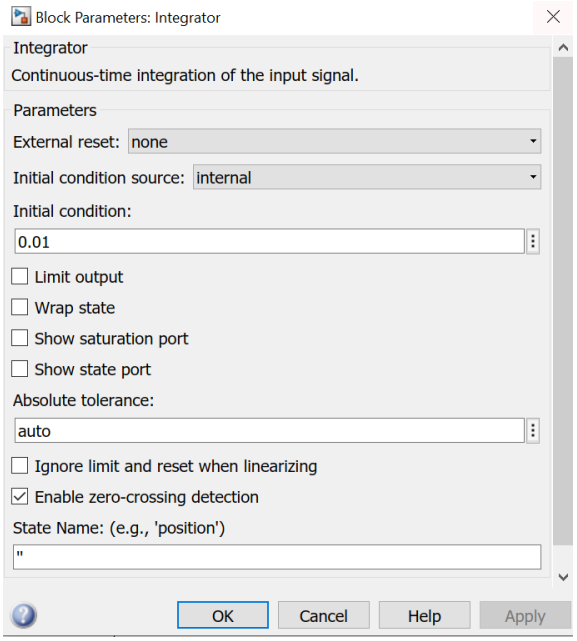

Figura n.29: Set dell'Integrator

Notiamo, dalle Figure n.28 e n.29 sopra, che sia il blocco State-Space che il blocco Integrator sono stati inizializzati a 0.01, in modo tale da poter tracciare l'andamento effettivo della velocità e della posizione verso il riferimento Yopt che è pari a 0.

Il PID è stato settato con i seguenti valori:

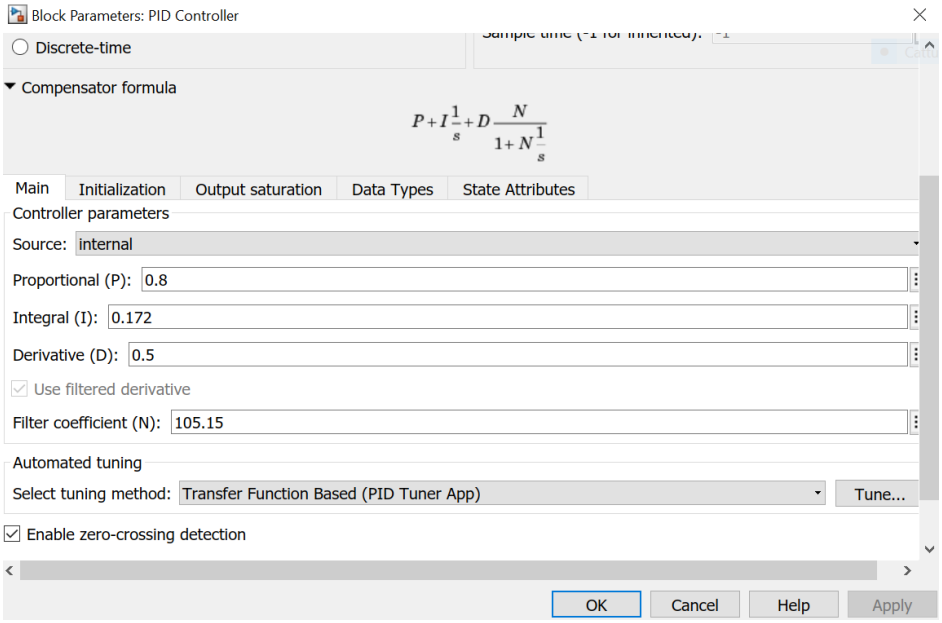

Figura n.30: Set del PID (2)

A seguito di numerose prove, sono stati ricavati i seguenti parametri per l'algoritmo CFDL-MFAC applicato alla funzione di trasferimento del pitch:

```
ro = 0.00001;
epsistima = 10^-15;
etastima = 0.00001;
phi11 = 0.00001;
mistima = 1;
lambda = 0.0001;

Tc = 0.001;
Yopt = 0;
```

Figura n.31: Parametri algoritmo CFDL per il drone

Che ci forniscono i seguenti grafici:

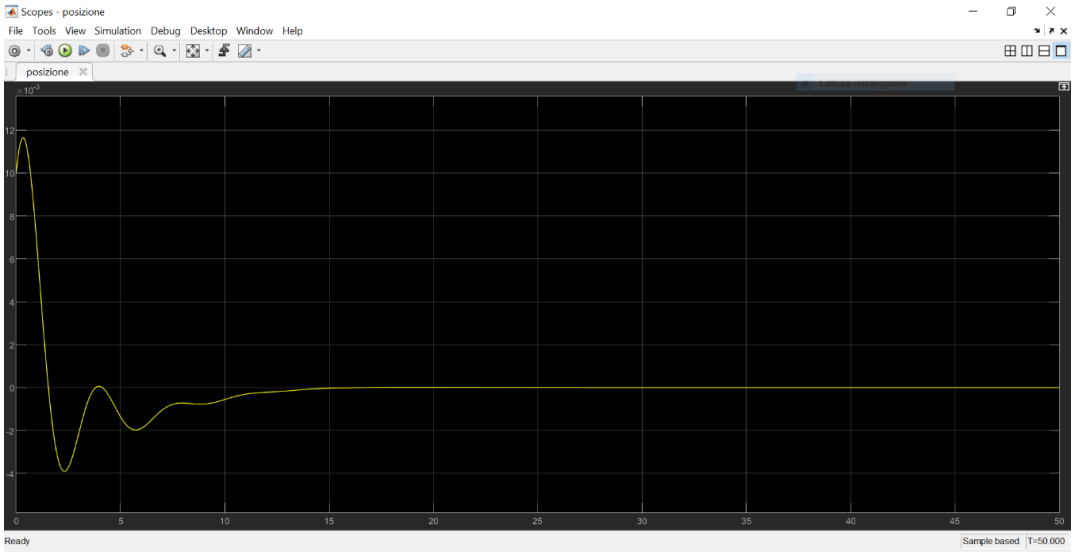

Figura n.32: Posizione del drone CFDL

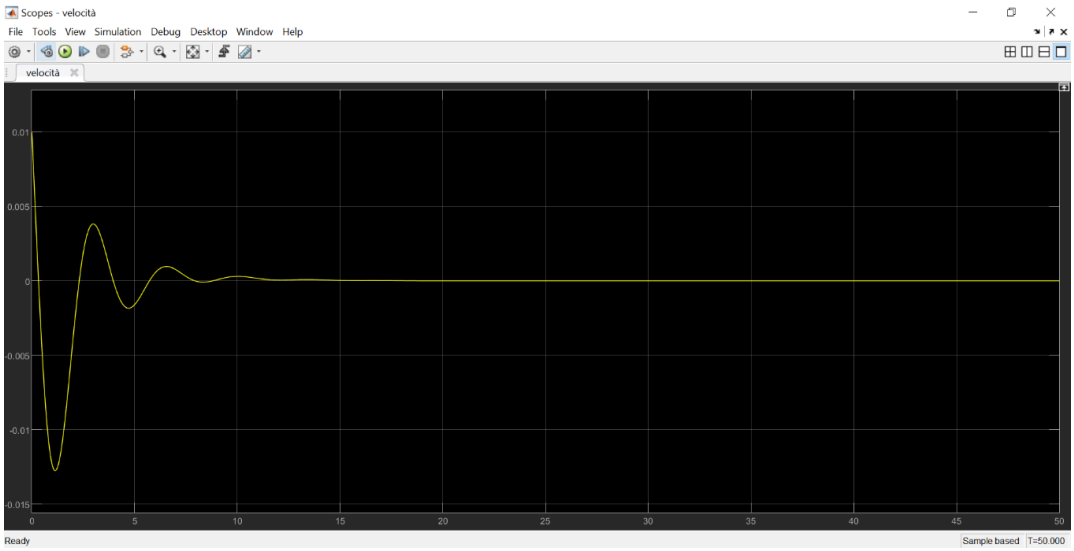

Figura n.33: Velocità del drone CFDL

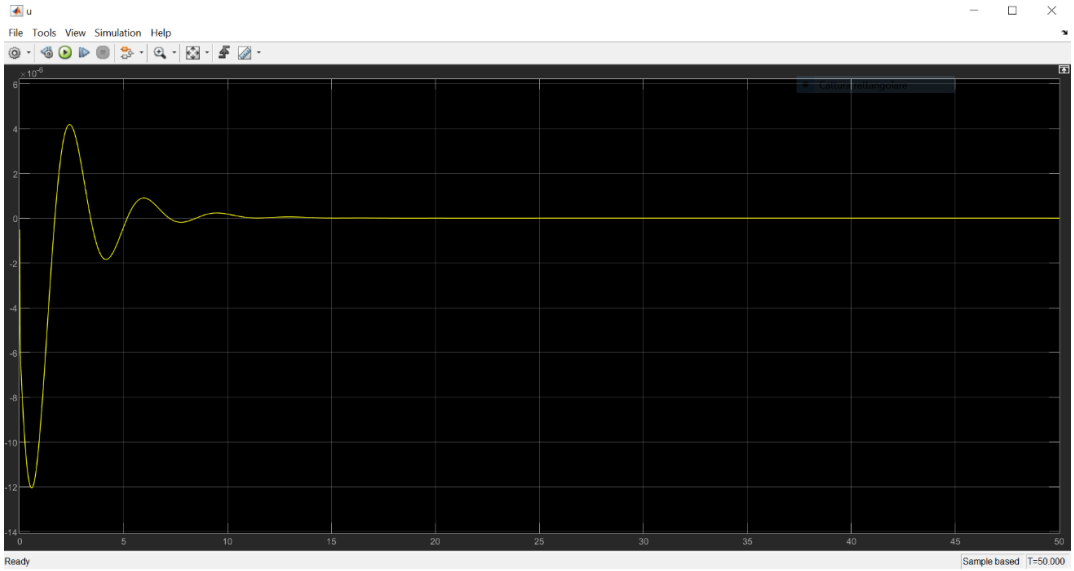

Figura n.34: Sforzo di controllo del drone CFDL

Figura n.35: Errore del drone CFDL

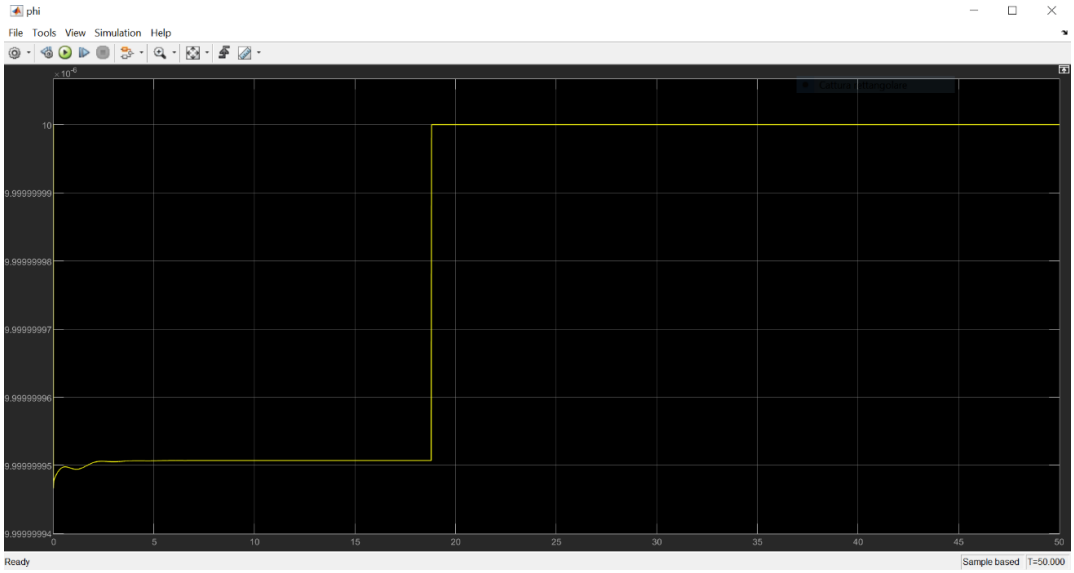

Figura n.36: Phi del drone CFDL

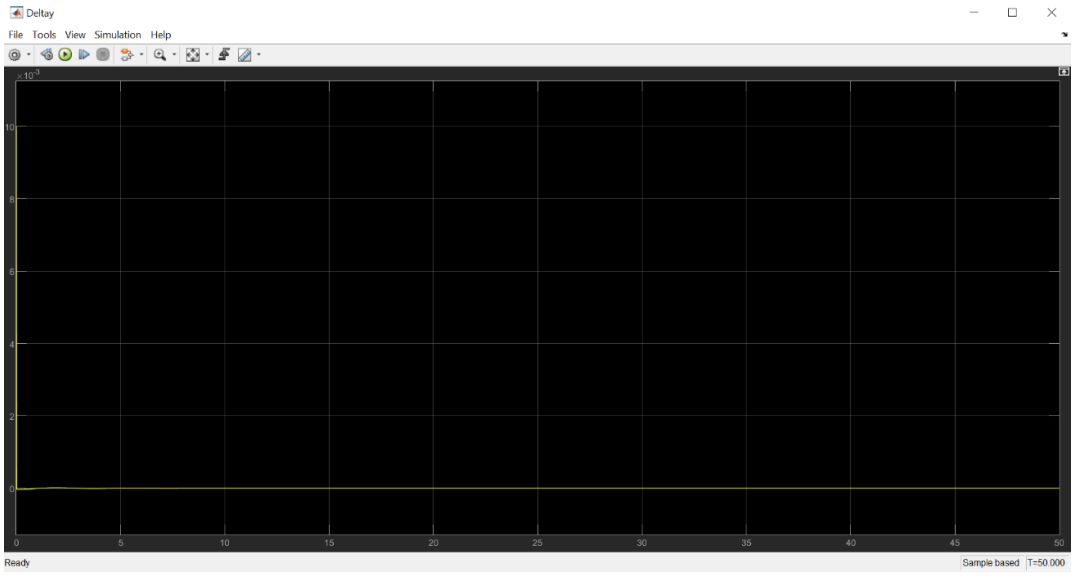

Figura n.37: Deltay del drone CFDL

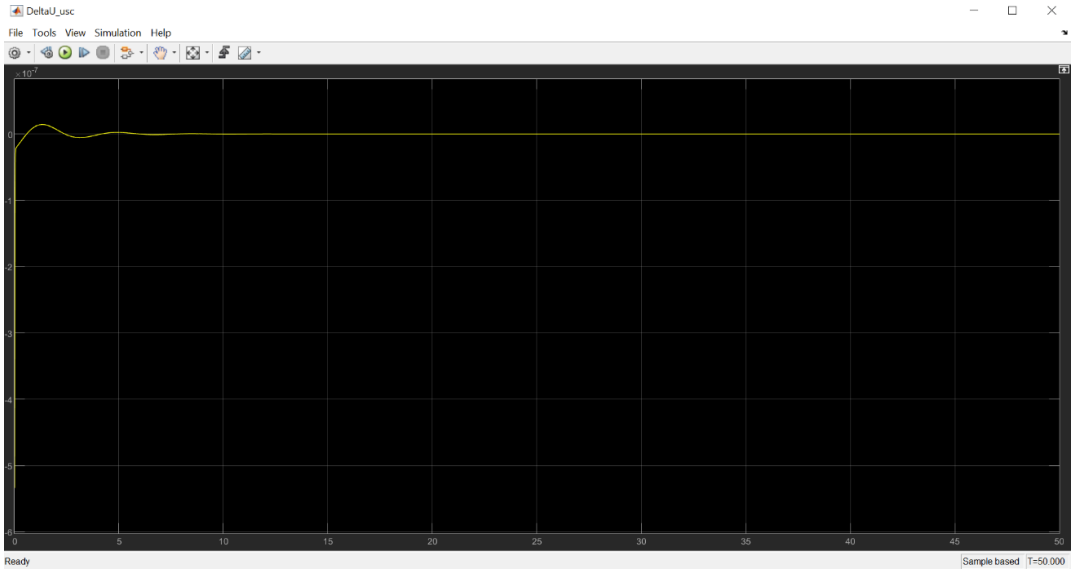

Figura n.38: DeltaU_usc del drone CFDL

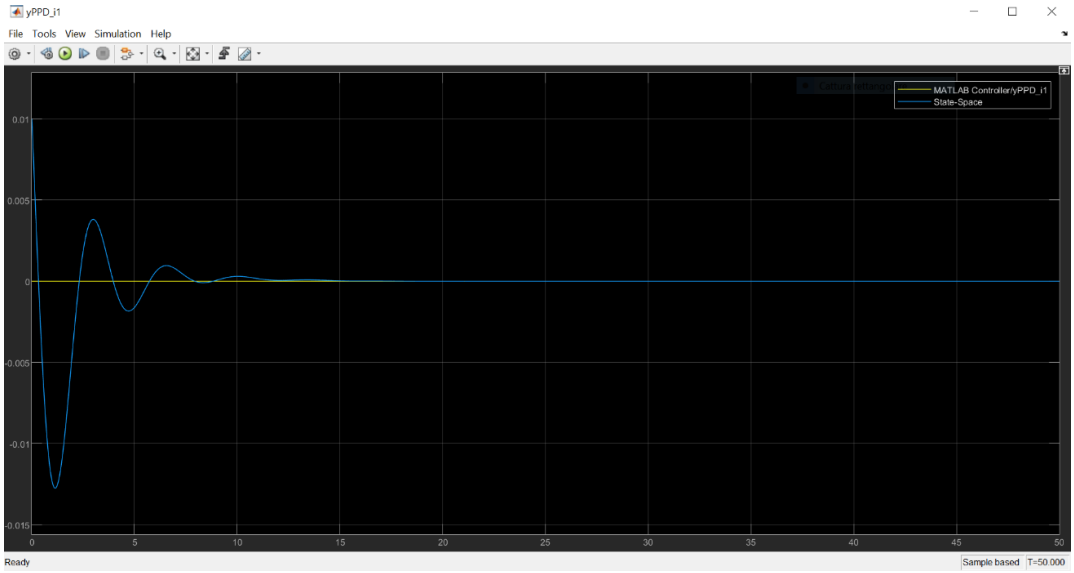

Figura n.39: Inseguimento stima del drone CFDL

Dalle figure sopra possiamo notare come l'output del controllore, ovvero lo sforzo di controllo u, sia oscillatorio nei primi secondi (dopodiché tende a stabilizzarsi) ed è sufficientemente piccolo, ma in grado di far stabilizzare entrambe posizione e velocità. La velocità, infatti, eredita per i primi secondi il comportamento oscillatorio della u (pur sempre cercando di seguire il comportamento stimato dal controllore) per poi stabilizzarsi dopo all'incirca una quindicina di secondi. Idem per la posizione, che tende ad oscillare per i primi 15 secondi e a stabilizzarsi in

seguito. Tutti i grafici sopra, sono accomunati da un comportamento più o meno oscillatorio per i primi secondi con sovraelongazione/sottoelongazione annessa per poi stabilizzarsi. I grafici ottenuti sono quindi il risultato della combinazione dei parametri inseriti, i quali ci consentono di ottenere un buon compromesso tra oscillazioni e velocità.

Nel corso degli esperimenti è stato notato che il parametro che ha maggiore influenza su questi risultati è ro (ρ). Per $\rho \geq 0.001$ si ha divergenza, mentre per $\rho \leq 0.000001$ si ha convergenza con un andamento sinusoidale piuttosto lento; per questo motivo la scelta è ricaduta tra i seguenti valori: $0.000001 < \rho < 0.001$. In fine è stato scelto $\rho = 0.00001$ per ottenere il giusto equilibrio tra tempo di salita e sottoelongazione. Per $\rho = 0.0001$ avremmo avuto una sottoelongazione minore ma un'oscillazione più persistente.

9.2 Controllo in Pitch: Simulazione PFDL-MFAC

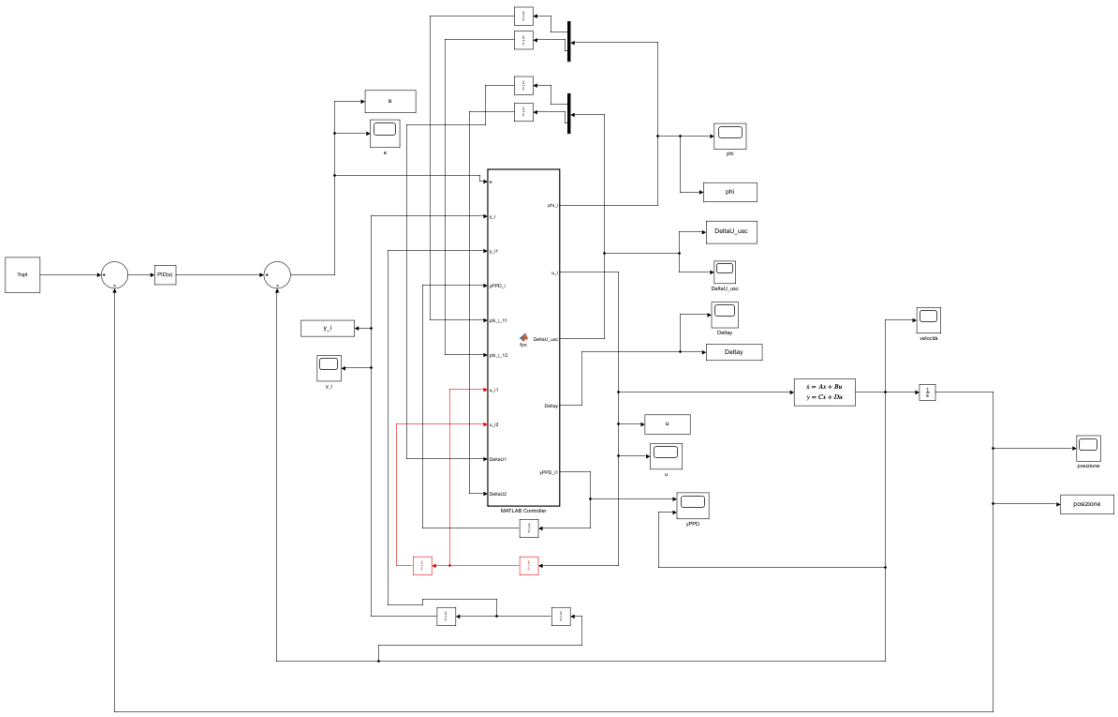

Figura n.40: Schema del drone PFDL

Come per la simulazione precedente, la velocità - ottenuta come output del blocco State-Space - viene integrata tramite un blocco Integrator al fine di ottenere la posizione. Anche qui sia il blocco State-Space che il blocco Integrator sono stati inizializzati a 0.01, per poter ottenere l'andamento effettivo della velocità e della posizione verso lo 0. Il PID è settato con i valori mostrati nella Figura n.30.

Dopo vari esperimenti, sono stati ricavati i seguenti parametri per l'algoritmo PFDL-MFAC applicato alla funzione di trasferimento del pitch:

```
ro = 0.00001;
ro1= 0.00001;
epsistima = 10^-15;
etastima = 0.00001;
phi11 = 0.00001;
phi12 = 0.00001;
mistima = 1;
lambda = 0.0001;

Tc = 0.001;
Yopt = 0;
```

Figura n.41: Parametri algoritmo PFDL per il drone

Che ci fanno ottenere i seguenti grafici:

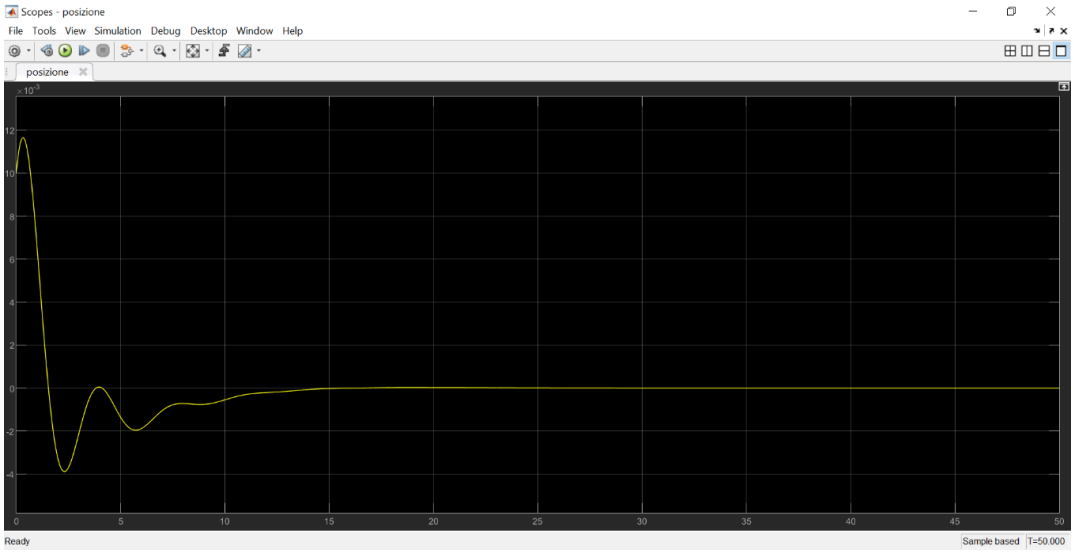

Figura n.42: Posizione del drone PFDL

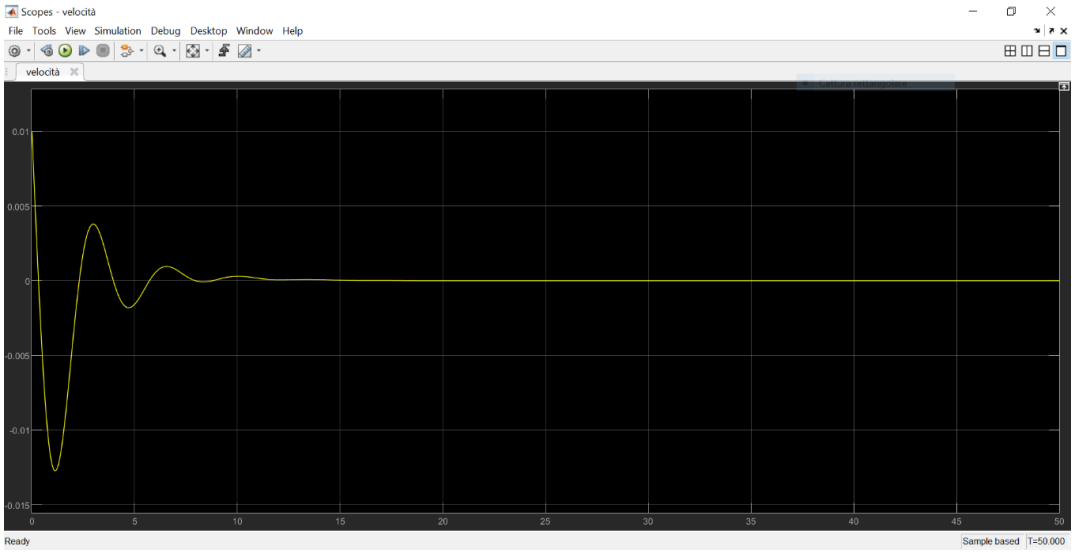

Figura n.43: Velocità del drone PFDL

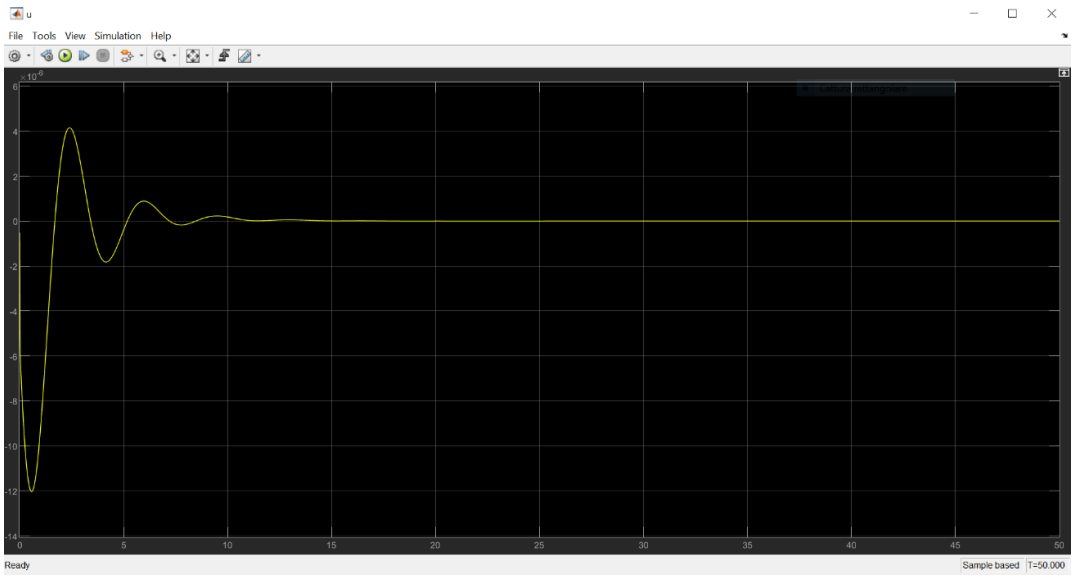

Figura n.44: Sforzo di controllo del drone PFDL

Figura n.45: Errore del drone PFDL

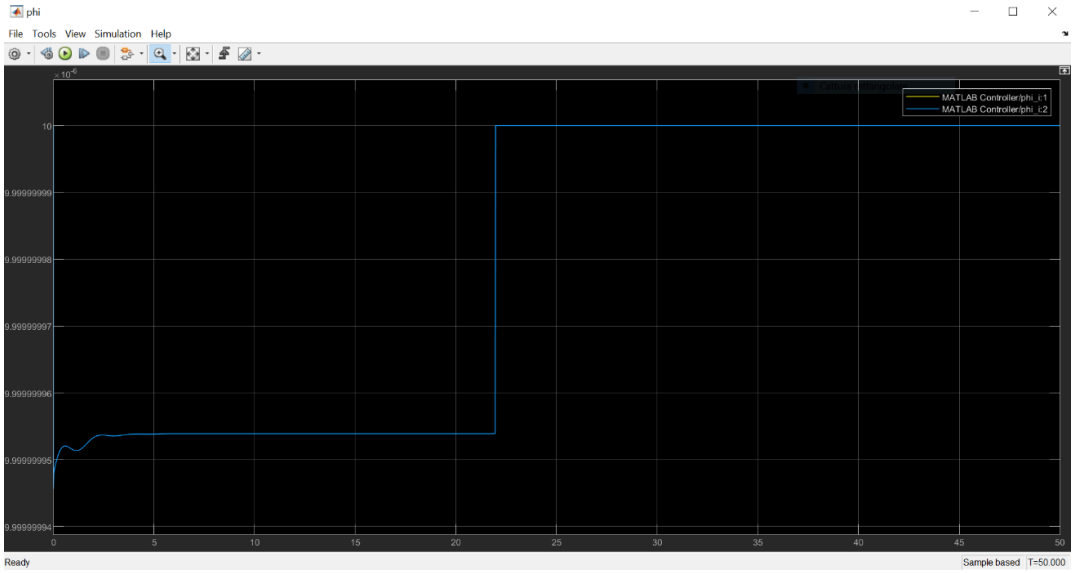

Figura n.46: Phi del drone PFDL

Figura n.47: DeltaU_usc del drone PFDL

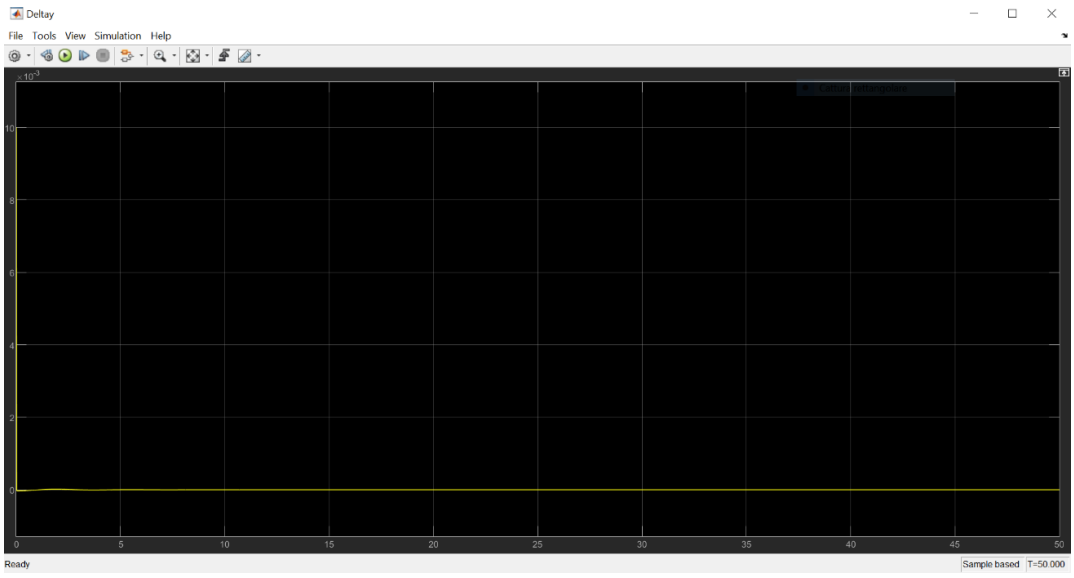

Figura n.48: Deltay del drone PFDL

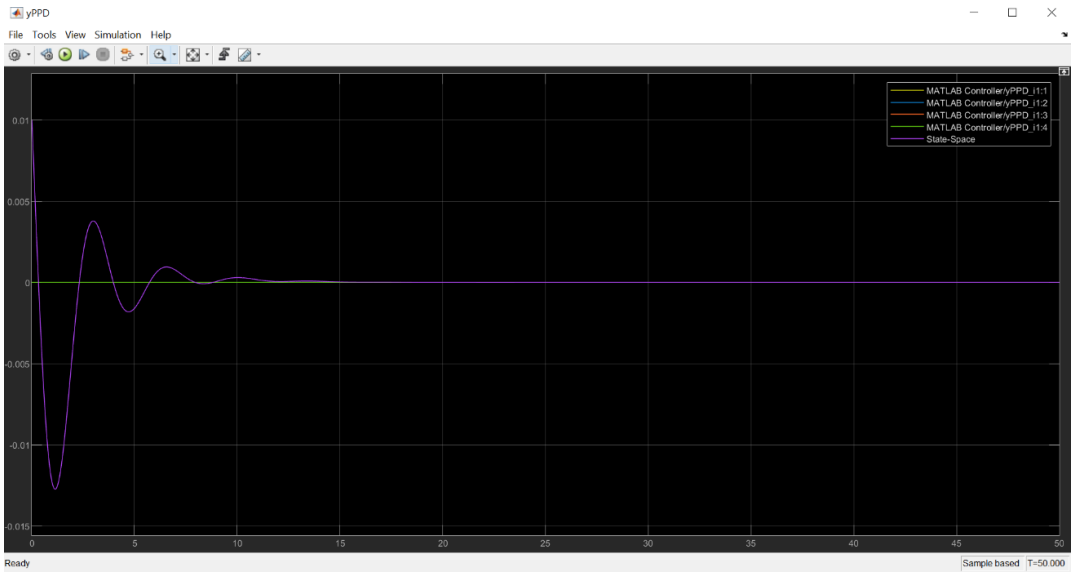

Figura n.49: Inseguimento della stima del drone PFDL

La prima osservazione importante da fare è che i parametri utilizzati per l'algoritmo CFDL-MFAC applicati al drone sono stati riproposti anche in questo caso. Ciò implica che i grafici ottenuti con l'algoritmo PFDL-MFAC sono identici a quelli ottenuti con l'algoritmo CFDL-MFAC. Infatti, anche qui lo sforzo di controllo (u), la velocità e la posizione sono oscillatorie per i primi 15 secondi, ma poi tendono a stabilizzarsi sul riferimento Yopt che è uguale a 0. In generale, anche in questo caso, il comportamento è oscillatorio per tutte le variabili nei primi secondi e si ha una sovraelongazione o sottoelongazione per poi stabilizzarsi.

Il parametro ro (ρ), anche qui, è il principale responsabile della variazione dei grafici. Infatti, come per il CFDL-MFAC, abbiamo ottenuto che l'intervallo di valori con cui ρ agisce in maniera migliore è il seguente: $0.000001 < \rho < 0.001$. La scelta è ricaduta sul valore $\rho = 0.00001$ poiché fornisce un buon bilanciamento tra tempo di salita e sottoelongazione. Per quanto riguarda il parametro ro1 (ρ_1), esso non è particolarmente influente sui grafici, una volta fissato ρ.

Conclusioni

Ricordiamo innanzitutto che lo scopo dell'elaborato era quello di applicare e analizzare le tecniche data driven CFDL-MFAC e PFDL-MFAC sulla funzione di trasferimento ottenuta dal sottosistema di pitch del drone. Dopo aver mostrato le applicazioni e alcune osservazioni ad esse correlate è possibile giungere ad alcune conclusioni.

Una prima conclusione vede sia il CFDL-MFAC che il PFDL-MFAC come controllori in grado di raggiungere il riferimento imposto, nonostante l'ingresso del processo sia variabile nel tempo, implicando che essi siano entrambi capaci di adattarsi in un breve periodo di tempo. Per la stessa ragione, essi possono essere definiti controllori robusti. Tali caratteristiche dipendono, però, dalla combinazione dei parametri scelti in fase di progettazione del controllore.

La scelta dei parametri è infatti il passaggio più importante nel design del controllore, in quanto un solo parametro errato potrebbe portare all'intero squilibrio del controllore e quindi alla divergenza. In più, i parametri incidono anche sulle prestazioni del controllore quali ad esempio: velocità, tempo di salita, oscillazioni, sovraelongazioni, robustezza e altro.

Inoltre, l'accuratezza della funzione di trasferimento ricavata ha comportato l'utilizzo degli stessi parametri per entrambi i sistemi di controllo. In tal caso, il CFDL-MFAC, nonostante abbia generalmente un comportamento peggiore rispetto al PFDL-MFAC, raggiunge gli stessi risultati e le stesse performance di quest'ultimo.

Tabella delle Variabili Simulink

Variabile	Simbolo	Variabile	Simbolo
y(k+1)*	*Yopt*	$\widehat{\varphi_c}(k)$	*phi_i*
y(k+1)	*y_i1*	$\widehat{\varphi_c}(k-1)$	*phi_i_1*
y(k)	*y_i*	$\widehat{\varphi_{p,1}}(1)$	*phi11*
u(k+1)	*u_i1*	$\widehat{\varphi_{p,2}}(1)$	*phi12*
u(k)	*u_i*	$\Delta u(k)$	*DeltaU*
ρ	*ro*	$\Delta y(k)$	*Deltay*
ρ_1	*ro1*	*Tempo*	*Tc*
η	*etastima*	λ	*Lambda*
μ	*mistima*	ε	*epsistima*

Tabella dei Simboli

v	Velocità lineare nel sistema solidale con il drone
ω *o* ω^b	Velocità angolare nel sistema solidale con il drone
ω_i	Velocità dell'i-esimo motore
m	Massa del drone
I	Vettore inerzia del drone
g	Accelerazione di gravità
ρ	Densità dell'aria
r	Raggio dei rotori
A	Area dei rotori
R_B^E	Matrice di trasformazione in tre dimensioni
α	Angolo di attacco del rotore
v_i	Velocità indotta attraverso il rotore
σ	Rigidità del rotore
θ_{tip}	Angolo geometrico tra il piano e la punta del rotore
a	Profilo alare
V_C	Velocità verticale
CdM	Centro di massa
d_i	Posizionamento dell'i-esimo rotore rispetto il CdM
d	Lunghezza del braccio
t_i	Spinta dell'i-esimo motore
q_i	Coppia dell'i-esimo motore
m_i	Momento torcente dell'i-esimo motore
$a_{1_{si}}$	Angolo di flapping longitudinale

$b_{1_{si}}$	Angolo di flapping laterale
C_T	Coefficiente di spinta
C_Q	Coefficiente di coppia
K_i	Coefficiente di drag i-esimo lato, i = 1, ..., 6
h	Altezza del drone

Riferimenti Bibliografici e Sitografia:

[1] Dong Liu and Guang-Hong Yang: *Data-Driven Adaptive Sliding Mode Control of Nonlinear Discrete-Time Systems With Prescribed Performance: IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, VOL. 49, NO. 12, DECEMBER 2019.*

[2] Zhongsheng Hou and Shangtai Jin: *Model Free Adaptative Control Theory and Applications.*

[3] Ugo Rosolia and Francesco Borrelli: *Learning Model Predictive Control for Iterative Tasks. A Data-Driven Control Framework: IEEE TRANSACTIONS ON AUTOMATIC CONTROL, VOL. 63, NO. 7, JULY 2018.*

[4] Zhong-Sheng Hou and Zhuo Wang*: From model-based control to data-driven control: Survey, classification and perspective.*

[5] Zhongsheng Hou and Yuanming Zhu: *Model based control and MFAC, which is better in simulation? 13th IFAC Symposium on Large Scale Complex Systems: Theory and Applications July 7-10, 2013. Shanghai, China.*

[6] Paul Pounds, Robert Mahony, Peter Corke: *Modelling and Control of a Quad-Rotor Robot.*

[7] Michele Del Duca: *Controllo di un Quadcopter.*

[8] Matteo Pantalone: *MODELLAZIONE E SIMULAZIONE DI UN QUADRICOTTERO MULTIROTORE.*

[9] Sumaila Musa: *Techniques for Quadcopter Modelling & Design: A Review.*

[10] Samir BOUABDALLAH: *Design and control of quadrotors with application to autonomous flying.*

[11] Teppo Luukkonen: *Modelling and control of quadcopter.*

[12] Vocabliario online Treccani: https://www.treccani.it/vocabolario/adattivo/