UNIVERSITÀ
POLITECNICA
DELLE MARCHE

Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione

STUDIO E SVILUPPO DI OSSERVATORI PER IL CONTROLLO SENSORLESS DI MOTORI BRUSHLESS

Study and Developement of Observers for Sensorless Control of Brushless Motors

Candidate:
Niccolò Tassoni

Advisor:
Prof. Gianluca Ippoliti

Coadvisor:
Prof. Giuseppe Orlando

Academic Year 2023-2024

Università Politecnica delle Marche
Facoltà di Ingegneria
Corso di Laurea in Ingegneria Informatica e dell'Automazione
Via Brecce Bianche – 60131 Ancona (AN), Italy

Introduzione

Questa tesi si propone di studiare e sviluppare un osservatore basato sulla tecnica Super-Twisting Sliding Mode (STSMO) per il controllo sensorless di motori brushless a magneti permanenti (PMSM). L'obiettivo principale è progettare un sistema capace di stimare con elevata precisione variabili fondamentali come la velocità e la posizione del rotore, eliminando la necessità di sensori e migliorando così l'efficienza e l'affidabilità del sistema. Rispetto agli osservatori tradizionali, il metodo Super-Twisting offre una significativa riduzione del fenomeno del chattering, aumentando la qualità della stima.
L'osservatore è stato integrato in un sistema di controllo che utilizza un controllo sliding mode (SMC) per la stima della velocità e un controllo PI per la regolazione delle correnti del motore.
L'intero progetto è stato implementato in ambiente MATLAB Simulink, utilizzando un simulatore appositamente sviluppato. Questo simulatore comprende modelli dettagliati del motore, dell'inverter e del sistema di controllo, consentendo di testare e ottimizzare l'osservatore in condizioni operative realistiche. Durante le simulazioni, le prestazioni dell'approccio proposto sono state analizzate sia in termini di accuratezza delle stime sia di robustezza del controllo, con particolare attenzione alla riduzione delle oscillazioni indesiderate nelle variabili stimate.
La validità del sistema sviluppato è stata verificata mediante simulazioni che hanno confrontato le prestazioni del controllo sensorless con quelle di un approccio tradizionale basato sulla lettura dei sensori.
Infine il lavoro complessivo è stato validato attraverso il kit di sviluppo MCSPTE1AK144, fornito dalla NXP, che ha permesso di utilizzare il codice sviluppato in un applicazione reale, e quindi di testare la coerenza tra la simulazione e la realtà.
La tesi è organizzata in 8 capitoli principali. Il Capitolo 1 descrive l'hardware utilizzato. Il Capitolo 2 introduce gli strumenti software impiegati, come MATLAB Simulink, MBDToolbox e FreeMASTER. Nel Capitolo 3 viene introdotto il simulatore, includendo dettagli sulla sua struttura e sui principali componenti. Il Capitolo 4 introduce la teoria preliminare alla base del progetto, mentre il Capitolo 5 approfondisce la teoria relativa all'osservatore, descrivendone il modello matematico, l'algoritmo Super-Twisting e il PLL utilizzato per la stima della velocità. Il Capitolo 6 si concentra sull'implementazione Simulink dell'osservatore. Il Capitolo 7 è dedicato alle simulazioni, descrivendo il tuning dell'osservatore, i test iniziali e le prove con l'osservatore integrato nella logica di controllo. Infine, nel capitolo 8 vengono validati i risultati ottenuti in simulazione attraverso il kit MCSPTE1AK144.

Contents

Introduzione 4

1 Hardware 5
1.1 MCSPTE1AK144 5

2 Software 7
2.1 Matlab - Simulink 7
2.2 MBDToolbox 7
2.3 FreeMASTER 8

3 Simulatore 9
3.1 Inizializzazione delle variabili e Lettura dei sensori 10
3.2 Controllo di velocità 11
3.3 Controllo delle correnti 12
3.4 Inverter e modello del motore 13
3.5 Diagramma di Flusso 14

4 Teoria e Nozioni preliminari 15
4.1 Trasformata di Park 15
4.1.1 Definizione matematica della trasformata di Park 15
4.2 Controllo Sliding Mode 17
4.2.1 Formulazione del Problema 17
4.2.2 Problemi e Caratteristiche 20
4.2.3 Scelta della superficie di sliding 21

5 Osservatore - Teoria 22
5.1 Modello Matematico dell'Osservatore 22
5.1.1 Algoritmo Super-Twisting Sliding Mode 23
5.1.2 Condizioni di Stabilità dell'Algoritmo 23
5.2 Stima delle Back-EMF 24
5.3 Modello Matematico del PLL 25

6 Osservatore - Implementazione 27
6.1 Input dell'Osservatore 27

6.2 Funzione Saturazione 28
6.2.1 Implementazione Matlab 28
6.3 Stima delle Back-EMF 30
6.3.1 Implementazione Matlab 31
6.4 Phase-Locked Loop 32
6.5 Osservatore inserito nel sistema di controllo delle correnti 32
6.5.1 Scaling 32

7 Simulazione **34**
7.1 Tuning dell'Osservatore 34
7.2 Test Iniziali 35
7.2.1 Ingresso a Gradino 35
7.2.2 Discussione dei risultati 38
7.2.3 Ingresso a Rampa 39
7.2.4 Discussione dei risultati 42
7.3 Osservatore Inserito nella Logica di Controllo 42
7.3.1 Riferimento a gradino 43
7.3.2 Riferimento a rampa 44
7.4 Discussione dei risultati 44

8 Validazione **45**

9 Conclusioni e Sviluppi futuri **46**

Chapter 1

Hardware

1.1 MCSPTE1AK144

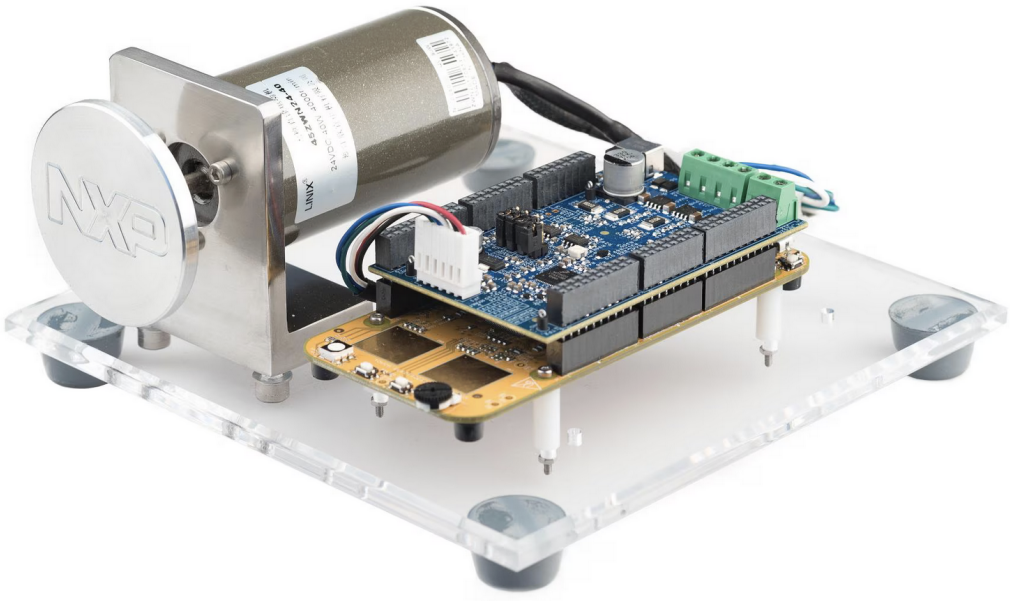

Figure 1.1: *MCSPTE1AK144*

L'hardware impiegato per lo sviluppo del progetto è il S32K144 Development Kit, progettato specificamente per il controllo di motori brushless a corrente continua (BLDC) e motori sincroni a magneti permanenti trifase (PMSM). Questo kit, identificato con la sigla MCSPTE1AK144, è particolarmente adatto per applicazioni industriali come sistemi HVAC (riscaldamento, ventilazione e condizionamento dell'aria), pompe elettriche, sospensioni attive e sistemi di trazione elettrica.

Il kit si basa sul microcontrollore S32K144, appartenente alla serie S32K1, che integra un core Arm® Cortex®-M a 32 bit, e sul pre-driver GD3000, progettato per pilotare inverter trifase con efficienza elevata. Questa combinazione con-

sente una gestione flessibile ed efficiente delle applicazioni di controllo motore, garantendo tempi di sviluppo ridotti e la possibilità di valutare rapidamente le prestazioni del sistema prima della definizione dell'hardware finale.

Un elemento distintivo del kit MCSPTE1AK144 è il supporto software, che sfrutta la Automotive Math and Motor Control Library Set per fornire un'implementazione di riferimento completa per il controllo di motori BLDC e PMSM. Questa libreria offre algoritmi avanzati che agevolano lo sviluppo e l'ottimizzazione delle strategie di controllo.

Grazie a queste caratteristiche, il kit si presenta come una soluzione completa e versatile per lo sviluppo di sistemi di controllo motore, riducendo significativamente la complessità associata alla progettazione hardware e software.

Il kit si compone di:

- S32K144 Evaluation Board
- DEVKIT-MOTORGD: Scheda di potenza trifase fino a 12 V/5 A basata sul predriver SMARTMOS™ GD3000 con monitoraggio delle condizioni e rilevamento dei fault.
- Motore a magneti permanenti tri fase con sensori HALL

Chapter 2

Software

I software utilizzati sono: Matlab - Simulink e FreeMASTER.

2.1 Matlab - Simulink

Simulink è un ambiente di diagrammi a blocchi utilizzato per progettare sistemi con modelli multidominio, effettuare simulazioni prima di passare all'hardware e procedere alla distribuzione senza scrivere codice. È integrato in MATLAB e offre una piattaforma flessibile per progettare, simulare e testare sistemi complessi. Mette a disposizione un'interfaccia grafica, che consente di rappresentare il comportamento di un sistema attraverso schemi a blocchi. Ogni blocco rappresenta una funzione o un componente del sistema. In questo modo semplifica il processo di modellazione, favorendo una comprensione intuitiva del sistema.

2.2 MBDToolbox

Il Model-Based Design Toolbox (MBDT) di NXP consente la prototipazione rapida di algoritmi complessi su hardware NXP direttamente da Simulink. Funziona come un ponte tra l'ecosistema MathWorks e le soluzioni software di NXP. Grazie all'integrazione con MathWorks, MBDT offre un ambiente di sviluppo applicativo incentrato sul modello, permettendo l'accesso a una suite di toolbox pronti all'uso, la simulazione di algoritmi indipendenti dall'hardware e la generazione automatica di codice da script MATLAB e modelli Simulink. Include inoltre supporto nativo per i flussi di verifica e validazione Software-in-the-Loop (SIL) e Processor-in-the-Loop (PIL).

MBDT integra tutte le risorse necessarie per personalizzare i progetti per le piattaforme NXP, collegando l'accesso hardware con una libreria di blocchi che generano codice basato su driver ottimizzati per piattaforme specifiche (ad esempio SDK, RTD, BMS SDK, RSDK). Permette la configurazione delle periferiche di bordo, dei pin e dei clock in ambienti dedicati come gli S32 Configuration Tools/MCUXpresso Configuration Tools di NXP e EB Tresos, sup-

portando l'uso di librerie ottimizzate come Automotive Math and Motor Control (AMMCLib), Real Time Control Embedded Software Motor Control and Power Conversion Libraries (RTCESL), librerie per acceleratori hardware (ad esempio SPF2) e risultati di simulazione bit-accurate nell'ambiente Simulink (ad esempio SPT).

Grazie al supporto di più opzioni di compilatori, MBDT semplifica il processo di simulazione, generazione del codice, compilazione e distribuzione automatica sui core hardware e acceleratori NXP. Questo accelera lo sviluppo di soluzioni complesse, come algoritmi per il controllo motore, sistemi di gestione delle batterie, applicazioni RADAR e protocolli basati su sensori e comunicazione, aiutando ingegneri e sviluppatori embedded a ridurre i tempi di sviluppo dei progetti.

2.3 FreeMASTER

Un altro strumento software utilizzato nel progetto è stato FreeMASTER, una piattaforma per il monitoraggio in tempo reale, il debug e la visualizzazione dei dati nelle applicazioni embedded. FreeMASTER si è rivelato particolarmente utile per verificare il funzionamento degli algoritmi sviluppati e per avviare il motore al di fuori dell'ambiente di simulazione, consentendo un passaggio graduale verso la validazione su hardware reale.

Grazie alla sua capacità di interfacciarsi direttamente con il microcontrollore utilizzato, FreeMASTER ha permesso di osservare in tempo reale le variabili chiave del sistema, come correnti, tensioni e velocità stimate. Questo ha facilitato l'identificazione di eventuali anomalie, l'ottimizzazione degli algoritmi di controllo e il confronto dei risultati con quelli ottenuti nelle simulazioni.

Chapter 3

Simulatore

Il simulatore utilizzato per questo progetto è basato su un modello Simulink che integra vari sottosistemi necessari per la simulazione del motore e il controllo associato. Il modello è strutturato in una sequenza logica che parte dalla ricezione dei segnali dai sensori e arriva alla simulazione del motore e del suo controllo.

Il flusso del modello inizia con un sottosistema di ricezione dei dati, che acquisisce le informazioni necessarie per il funzionamento del sistema. Successivamente, i segnali vengono elaborati da un controllore di velocità e da un controllore di corrente, che regolano il comportamento del motore in base agli obiettivi di controllo. Il comportamento del sistema è rappresentato dal modello dell'inverter e del motore, che simula il comportamento fisico del motore sincrono a magneti permanenti e le dinamiche di potenza necessarie per la sua operazione.

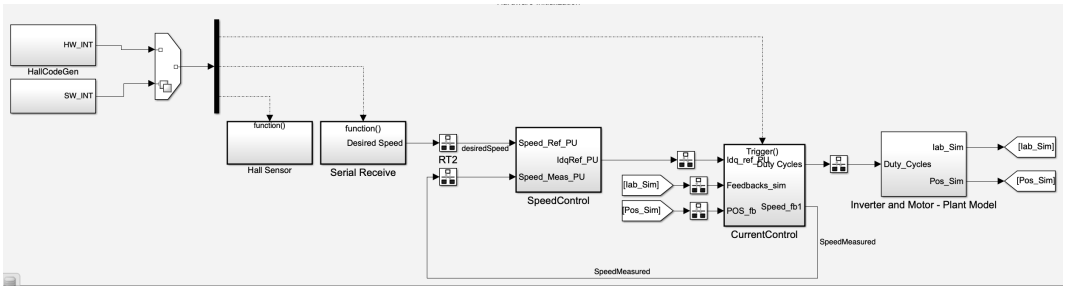

Figure 3.1: *Schema a blocchi Simulink*

3.1 Inizializzazione delle variabili e Lettura dei sensori

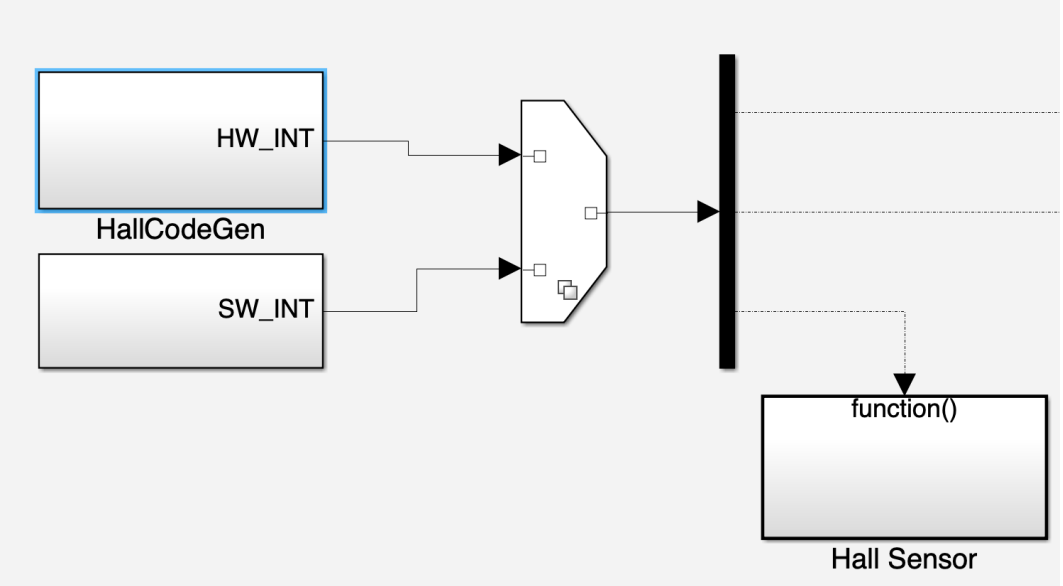

Figure 3.2: *INIT e Hall sensor*

Partendo da sinistra i primi blocchi hanno il compito di modificare il comportamento del programma nel caso in cui si tratta di una simulazione o stia caricando il codice all'interno della scheda.
Il sottosistema Hall Sensor serve a verificare la coerenza dei valori misurati dai sensori a effetto Hall e assicurare la comunicazione tra la board e il motore.

3.2 Controllo di velocità

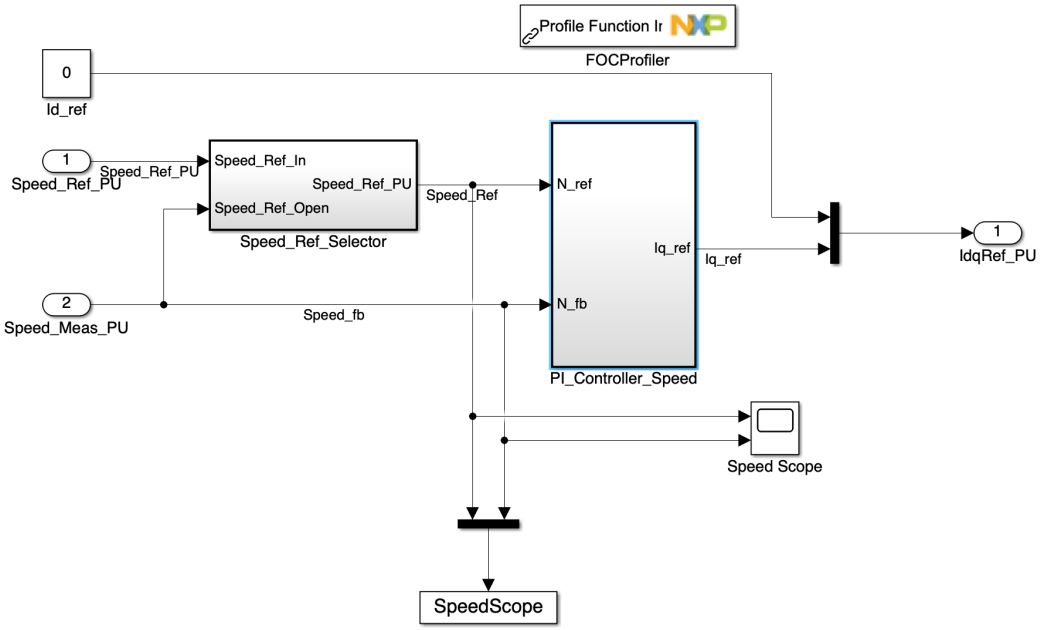

Figure 3.3: *Controllo della Velocità*

Il sottosistema di controllo della velocità è responsabile del mantenimento del comportamento desiderato del motore. Esso riceve in ingresso la velocità di riferimento e la velocità di retroazione, viene poi calcolata la loro differenza per ottenere l'errore di velocità, che deve essere minimizzato per garantire che il sistema abbia le prestazioni richieste.

Il controllo dell'errore di velocità è realizzato tramite un controllore PI o un controllore in sliding mode, a seconda delle specifiche esigenze del sistema.
L'output del sottosistema è il riferimento delle correnti espressi sugli assi d e q, che verranno poi utilizzati nel sottosistema di controllo della corrente per regolare la potenza fornita al motore e ottenere il comportamento desiderato.

3.3 Controllo delle correnti

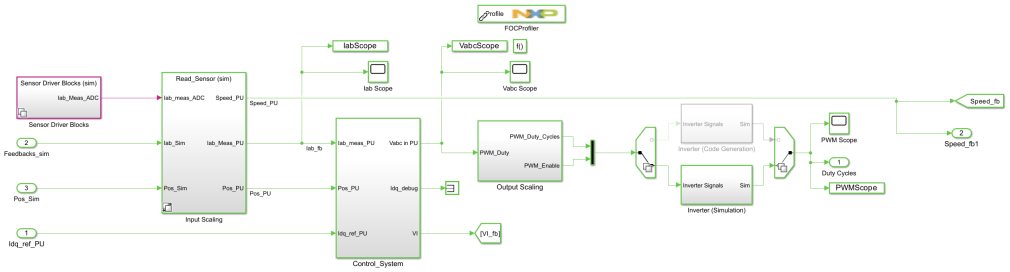

Figure 3.4: *Controllo delle correnti*

Inizialmente viene effettuata la lettura dei sensori e viene convertita la posizione dall'angolo meccanico misurato all'angolo elettrico θ_e, viene inoltre calcolata la velocità del motore tramite una derivata. Si passa poi all'effettivo sistema di controllo delle correnti.

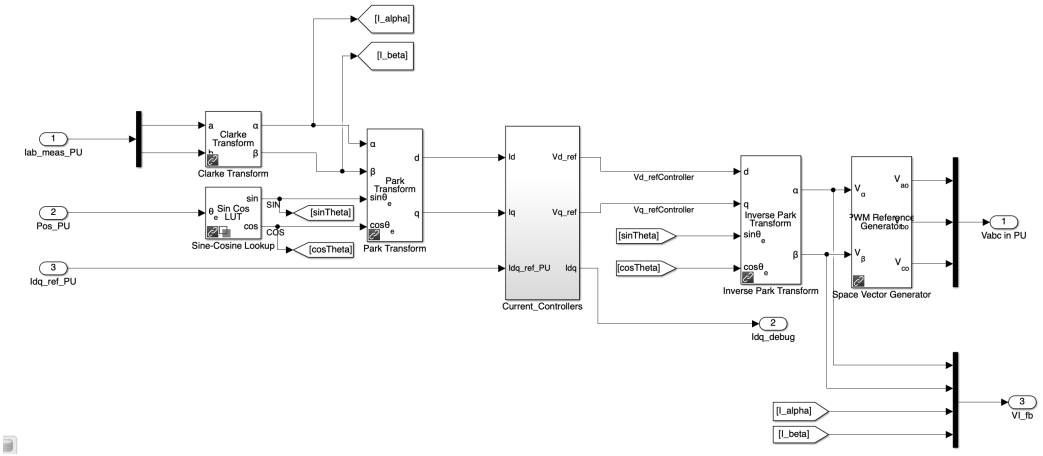

Figure 3.5: *Sistema di controllo delle correnti*

Il sottosistema dedicato al controllo delle correnti regola le tensioni da inviare al motore affinché questo segua la velocità desiderata, mantenendole entro un range accettabile per il suo funzionamento. Gli ingressi del sottosistema includono le correnti misurate, la posizione del rotore e i valori di riferimento delle correnti.

Poichè il controllo delle correnti è progettato per gestire le correnti del motore nel sistema di riferimento d-q, è necessario adattare le correnti misurate dal motore a questo riferimento. A tal fine viene inizialmente utilizzata una trasformata di Clarke per ottenere le correnti espresse sul riferimento $\alpha - \beta$, dopodichè vengono calcolati il seno e il coseno dell'attuale posizione del rotore,

infine viene applicata la trasformata di Park in modo da ricavare le correnti espresse sul riferimento d-q.

Il controllo è realizzato da due controller PI indipendenti, uno per ciascun asse del sistema d-q, che calcolano le tensioni di controllo in base al confronto tra le correnti misurate e i valori di riferimento. Gli output dei controllori vengono quindi riconvertiti nel riferimento $\alpha - \beta$ tramite una trasformata inversa di Park e inviati come segnali di controllo all' inverter del motore.

3.4 Inverter e modello del motore

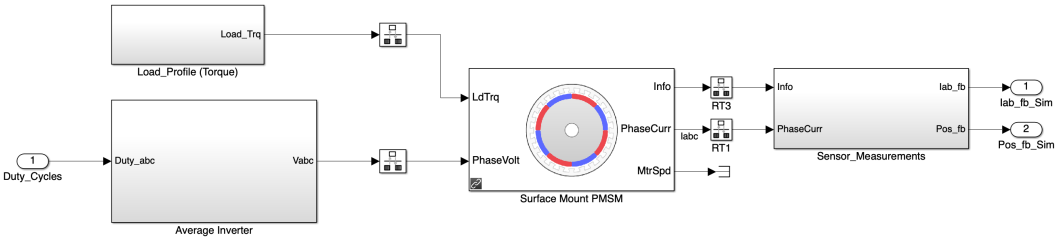

Figure 3.6: *Inverter e Motore*

Infine l'ultimo componente del sistema è riservato alla simulazione del motore e dell'inverter.
Il sottosistema ha il compito di simulare il comportamento fisico del motore e delle sue interazioni con il sistema di controllo.
L'inverter è modellato per generare le tensioni da applicare al motore, partendo dai segnali di controllo forniti dal sistema di regolazione.
Il modello del motore utilizzato è quello di un motore sincrono a magneti permanenti (PMSM). Questo modello include le equazioni dinamiche che governano il comportamento delle correnti, delle tensioni e della velocità angolare.
Il modello riceve in ingresso le tensioni generate dall'inverter e restituisce come output le correnti nei tre avvolgimenti statorici, la velocità e la posizione angolare del rotore.

3.5 Diagramma di Flusso

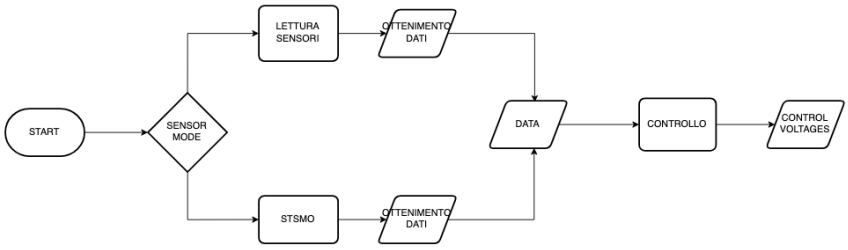

Figure 3.7: *Diagramma di Flusso*

Il diagramma di flusso in figura 3.7 mostra il funzionamento del sistema di raccolta dei dati utilizzato all'interno del simulatore. È possibile selezionare la modalità di acquisizione dei dati tramite degli interruttori, scegliendo tra l'uso dei sensori ad effetto Hall del motore o l'impiego dell'osservatore. Indipendentemente dalla modalità scelta, i dati così ottenuti vengono inviati all'unità di controllo, la quale genererà le tensioni da applicare al motore.

Chapter 4

Teoria e Nozioni preliminari

In questo capitolo verranno presentati e approfonditi alcuni concetti teorici alla base del progetto.

4.1 Trasformata di Park

La Trasformata di Park è uno strumento matematico utilizzato nell'ingegneria elettrica, in particolare nell'analisi delle macchine elettriche trifase come i motori a induzione e i generatori sincroni. Il suo scopo principale è semplificare l'analisi dei circuiti in corrente alternata (AC), soprattutto nel contesto della modellazione e del controllo delle macchine, trasformando grandezze trifase variabili nel tempo in un sistema di riferimento rotante, in cui le grandezze diventano invarianti nel tempo in condizioni di regime stazionario. Per semplificare questo processo, R.H. Park ha sviluppato una trasformazione che ha reso l'analisi delle macchine elettriche più agevole, convertendo le equazioni del motore in un sistema di riferimento che ruota in sincronia con i campi all'interno della macchina [7].
La trasformata opera partendo dal riferimento stazionario abc, che rappresenta le tensioni o le correnti delle tre fasi (a, b, c), e lo trasforma in un sistema rotante rispetto all'angolo di riferimento (in genere la posizione del rotore o del flusso dello statore) $dq0$. Il riferimento ottenuto consiste in:

- $asse - d(direct\ axis)$: allineato con il riferimento rotante, in genere associato con il flusso.
- $asse - q(quadrature\ axis)$: ortogonale rispetto all'asse d, rappresenta i componenti associati alla coppia.
- $asse - 0(zero - sequence\ component)$: rappresenta l'elemento della sequenza zero, che in genere è nullo nei sistemi equilibrati.

4.1.1 Definizione matematica della trasformata di Park

La trasformata di Park si compone di due passaggi: innanzitutto, le componenti delle tre fasi vengono trasformate in un sistema bifase stazionario uti-

lizzando la trasformata di Clarke, e successivamente vengono convertite in un sistema di riferimento rotante tramite la trasformata di Park vera e propria. Le variabili trifase v_a, v_b, v_c vengono trasformate in due componenti in un sistema di riferimento ortogonale stazionario v_α, v_β utilizzando la trasformata di Clarke:

$$\begin{pmatrix} v_\alpha \\ v_\beta \end{pmatrix} = \frac{2}{3} \begin{pmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{pmatrix} \begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix} \tag{4.1}$$

Dove:

- v_a, v_b, v_c sono le tensioni o le correnti trifase.
- v_α, v_β sono le componenti ortogonali nel sistema stazionario.

Successivamente il sistema di riferimento stazionario v_α, v_β viene trasformato in un sistema rotante dq. Questa trasformazione si basa sull'angolo elettrico θ, che definisce la rotazione del sistema di riferimento, tipicamente sincronizzata con il rotore della macchina o con il flusso statorico:

$$\begin{pmatrix} v_d \\ v_q \end{pmatrix} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} v_\alpha \\ v_\beta \end{pmatrix} \tag{4.2}$$

Dove:

- v_d e v_q sono le componenti nel sistema di riferimento rotante.
- θ è la posizione angolare del sistema di riferimento rotante, tipicamente l'angolo del rotore o l'angolo del flusso statorico.

Infine, nei sistemi trifase equilibrati, la componente di sequenza zero è nulla. Tuttavia, in presenza di condizioni squilibrate o asimmetrie, la componente di sequenza zero v_0 può essere calcolata come:

$$v_0 = \frac{1}{3}(v_a + v_b + v_c) \tag{4.3}$$

Pertanto, la trasformata completa di Park, che include la componente di sequenza zero, è data da:

$$\begin{pmatrix} v_d \\ v_q \\ v_0 \end{pmatrix} = \frac{2}{3} \begin{pmatrix} \cos(\theta) & \cos\left(\theta - \frac{2\pi}{3}\right) & \cos\left(\theta + \frac{2\pi}{3}\right) \\ \sin(\theta) & \sin\left(\theta - \frac{2\pi}{3}\right) & \sin\left(\theta + \frac{2\pi}{3}\right) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{pmatrix} \begin{pmatrix} v_a \\ v_b \\ v_c \end{pmatrix} \tag{4.4}$$

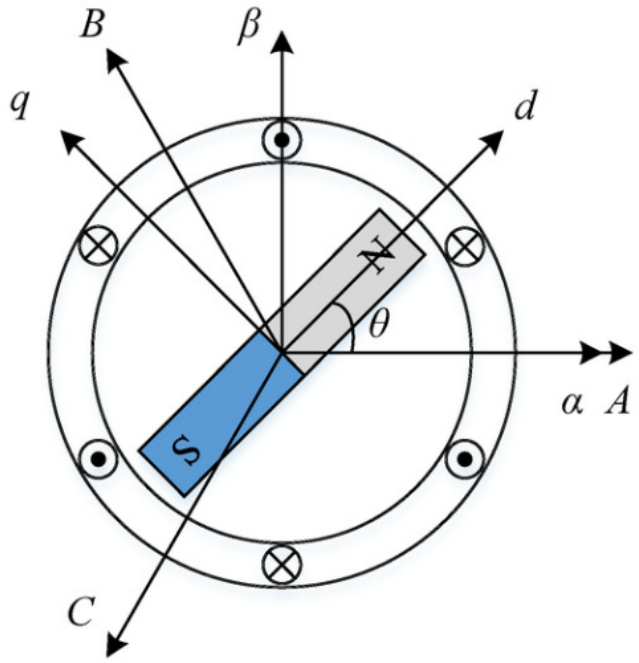

Figure 4.1: *Assi di riferimento*

4.2 Controllo Sliding Mode

Il controllo a sliding mode (SMC) è una strategia di controllo non lineare, progettata per portare e mantenere un sistema su una superficie di scorrimento (Sliding Surface), dove la dinamica del sistema è insensibile a determinate perturbazioni e incertezze dei parametri. La legge di controllo è composta da un controllo equivalente e un controllo discontinuo per garantire robustezza e convergenza in tempo finito[6], [1].

4.2.1 Formulazione del Problema

Dato il sistema descritto dalla seguente equazione differenziale vettoriale:

$$\dot{x} = f(x, u, t) \tag{4.5}$$

dove $x, f \in \mathbb{R}^n$, $u \in \mathbb{R}^m$, $t \in \mathbb{R}^+$. È inoltre fornita l'equazione vettoriale:

$$s(x) = 0 \tag{4.6}$$

Con $s(x) : \mathbb{R}^n \to \mathbb{R}^m$, cioè $s(x) = \Big[s_1(x) \dots s_m(x)\Big]^T$; si assume che le m equazioni scalari $s_i(x) = 0$ siano linearmente indipendenti.
Il problema consiste nel cercare una legge di controllo $u(x,t) = \Big[u_1(x,t) \dots u_m(x,t)\Big]^T$ della forma:

$$u_i(x,t) = \begin{cases} u_i^+(x,t) & \text{se } s_i(x) > 0 \\ u_i^-(x,t) & \text{se } s_i(x) < 0 \end{cases} \tag{4.7}$$

dove ($i = 1, \dots, m$, e $u_i^+(x,t)$, $u_i^-(x,t)$ sono funzioni continue, con $u_i^+(x,t) \neq u_i^-(x,t)$ quando $s_i(x) = 0$). L'obiettivo è che, a partire da un certo istante t_s, la condizione $s(x) = 0$ sia verificata, ossia lo stato del sistema si muove lungo l'intersezione delle m superfici.
Lo sviluppo della legge di controllo avviene in due fasi distinte: prima, si sceglie l'equazione $s(x) = 0$, selezionando una regione di dimensione $(n - m)$ in cui si desidera che avvenga la sliding mode. La scelta di questa regione è legata ai requisiti del sistema. Successivamente, le funzioni di controllo $u_i(x,t)$, con $i = 1, \dots, m$, devono essere selezionate per soddisfare l'equazione (4.6) a partire da t_s.

Considerando un sistema con un input di controllo scalare ($m = 1$) e $s(x) : \mathbb{R}^n \to \mathbb{R}$, queste sono le condizioni locali di sliding mode per l'esistenza:

$$\lim_{s \to 0^-} \dot{s} > 0, \quad \lim_{s \to 0^+} \dot{s} < 0 \quad \Leftrightarrow \quad \lim_{s \to 0} s\dot{s} < 0 \tag{4.8}$$

Esse garantiscono che il sliding mode si verifichi quando gli stati iniziali sono vicini alla superficie di sliding. Il metodo di Lyapunov è un'alternativa valida per definire le condizioni scalari per l'esistenza: un'espressione equivalente per la condizione nell'equazione (4.8) è la funzione definita positiva $V(s(x)) = \frac{1}{2}s^2(x)$ e la sua derivata definita negativa.
In generale, quando $u \in \mathbb{R}^m$, la seguente funzione di Lyapunov definita positiva è utilizzata:

$$V(s(x)) = \frac{1}{2}s^T(x)s(x) \tag{4.9}$$

con la sua derivata definita negativa:

$$\dot{V}(s(x)) = s^T(x)\dot{s}(x) < 0 \tag{4.10}$$

e una funzione di controllo della stessa forma dell'equazione (4.7). La funzione di controllo deve soddisfare la disuguaglianza (4.10) per ogni x. Per definire l'equazione di sliding mode, si utilizza la teoria delle equazioni differenziali con termini a destra discontinui di Filippov:

$$\dot{x} = f(x,t) + B(x,t)u(x,t) \tag{4.11}$$

ma questa forma è corretta solo per sistemi con controllo lineare. Dunque, viene adottato il metodo del controllo equivalente:

$$\frac{\partial s}{\partial x}\dot{x} = 0 \tag{4.12}$$

dove $\frac{\partial s}{\partial x}$ è il Jacobiano di $s(x)$. Sostituendo l'equazione (4.11) nell'equazione (4.12):

$$\frac{\partial s}{\partial x}f(x,t) + \frac{\partial s}{\partial x}B(x,t)u(x,t) = 0 \tag{4.13}$$

Esplicitando u e assumendo che la matrice $\left[\frac{\partial s}{\partial x}B(x,t)\right]$ abbia l'inverso, otteniamo:

$$u_{eq} = -\left[\frac{\partial s}{\partial x}B(x,t)\right]^{-1}\frac{\partial s}{\partial x}f(x,t) \tag{4.14}$$

La funzione u_{eq} è il controllo continuo equivalente e sostituendola nell'equazione (4.11), si ottiene l'equazione ideale di sliding mode:

$$\dot{x} = \left\{I - B(x,t)\left[\frac{\partial s}{\partial x}B(x,t)\right]^{-1}\frac{\partial s}{\partial x}\right\}f(x,t) \tag{4.15}$$

dove I è la matrice identità di ordine n. Se $\frac{\partial s}{\partial x}B(x,t)$ è singolare:

- Il controllo equivalente non è unico, ma l'equazione di sliding mode continua ad essere unica;
- Oppure il controllo equivalente non esiste, quindi non si verificano movi-

menti di sliding dello stato.

Lo sliding mode è invariato alle trasformazioni non-singolari della superficie di sliding o del vettore di controllo; data questa trasformazione:

$$s^*(x) = H_s(x,t)s(x) \tag{4.16}$$

dove $H_s(x,t)$ è una matrice $m \times m$ con $\det[H_s(x,t)] \neq 0$, l'equazione di sliding mode è ancora valida se i m componenti del vettore di controllo sono discontinui sulle superfici $s_i^*(x) = 0$, con $i = 1, \ldots, m$. Definendo anche

$$u^*(x) = H_u(x,t)u(x) \tag{4.17}$$

dove $H_u(x,t)$ è una matrice $m \times m$ con $\det[H_u(x,t)] \neq 0$, l'equazione di sliding non cambia se le m componenti del nuovo vettore di controllo $u_i^*(x)$ sono discontinui sulle superfici $s_i^*(x) = 0$. Le ultime due equazioni (4.16, 4.17) consentono di scegliere i m componenti del vettore di controllo indipendentemente. Pertanto, il problema può essere risolto come problemi monodimensionali.

4.2.2 Problemi e Caratteristiche

Lo scorrimento lungo la superficie rappresenta una condizione ideale; tuttavia, nella pratica, si verificano oscillazioni della traiettoria nelle vicinanze di $s(x) = 0$ a causa del meccanismo di commutazione non ideale, che non è fisicamente realizzabile a causa della frequenza di commutazione infinita che richiederebbe. Il ritardo introdotto dalla frequenza di commutazione porta anche al fatto che lo stato non si allinei prontamente con la superficie di sliding, portando al fenomeno del chattering: la generazione di oscillazioni ad alta frequenza causate dalla commutazione. Il meccanismo di commutazione può essere caratterizzato da un ciclo di isteresi di ampiezza Δ, dove $s(x) = 0$ è considerato scalare.
A causa del comportamento di $u(x,t)$, le traiettorie che partono da un punto in cui $s(x) < 0$ non rimangono nei punti dove $s(x) = 0$; invece, si spostano a $s(x) = +\Delta$ e poi ritornano a $s(x) = -\Delta$, e così via, generando oscillazioni. È evidente che valori più piccoli di Δ portano a una frequenza più alta e a una minore ampiezza di chattering. Quando $\Delta = 0$ (commutazione ideale), la frequenza sarebbe infinita e l'ampiezza di chattering sarebbe zero; ciò implica che lo stato "scivolerebbe" perfettamente sulla superficie $s(x) = 0$.
Richiedere che lo stato sia vicino alla superficie di sliding, piuttosto che ottenere

un moto di sliding perfetto, può aiutare a ridurre il fenomeno del chattering. Considerando il caso scalare:

$$|s(x)| \leq \varepsilon \tag{4.18}$$

Viene definita un'area chiamata "Boundary layer", con $\varepsilon > 0$. Fuori da questo strato, la funzione di controllo è definita come:

$$u(x,t) = \begin{cases} u^+(x,t) & \text{se } s(x) > \varepsilon \\ u^-(x,t) & \text{se } s(x) < -\varepsilon \end{cases} \tag{4.19}$$

Questa definizione assicura che lo stato raggiunga lo strato di confine e vi rimanga. All'interno dello strato di confine, la funzione di controllo è definita come:

$$u(x,t) = \begin{cases} k \cdot \text{sign}[s(x)] & \text{se } |s(x)| > \varepsilon \\ k \cdot \frac{s(x)}{\varepsilon} & \text{se } |s(x)| \leq \varepsilon \end{cases} \tag{3.25}$$

Utilizzando questo metodo, la legge di controllo è influenzata dal valore di ε, che deve essere scelto per ottimizzare l'equilibrio tra minimizzare il chattering e soddisfare i requisiti di controllo.

4.2.3 Scelta della superficie di sliding

La superficie di scorrimento dipende dal tempo. La selezione della superficie di scorrimento definisce geometricamente i requisiti di controllo. Nel contesto di un problema di tracking, si assume che l'uscita del sistema sia $y = h(x)$, dove $y, h \in \mathbb{R}^m$. Sia y_d l'uscita di riferimento e $e = y - y_d$ l'errore. La superficie di scorrimento è definita come segue:

$$s(x,t) = \dot{e} + \Lambda e = 0 \quad \text{con } \Lambda = \text{diag}(\lambda_i),\ \lambda_i > 0,\ i = 1, \ldots, m \tag{4.20}$$

Quando l'equazione è soddisfatta, ogni componente dell'errore e_i si avvicina esponenzialmente a zero con una costante di tempo pari a $\frac{1}{\lambda_i}$.

Chapter 5

Osservatore - Teoria

In questa tesi è stato implementato un Osservatore di stato Super-Twisting Sliding Mode (STSMO), assieme ad un Phase-locked Loop (PLL). Il STSMO è una variante della tecnica Sliding Mode, che cerca di superare le limitazioni di un osservatore di primo ordine [8]. Uno dei principali obiettivi è quello di limitare il fenomeno del Chattering, che può comportare instabilità e amplificare il rumore.

In questo progetto l'osservatore è stato utilizzato per stimare le Back-EMF del motore, viene poi utilizzato il PLL per individuare la posizione del rotore e la velocità del motore.

5.1 Modello Matematico dell'Osservatore

Il modello matematico di un motore PMSM rispetto al riferimento $\alpha - \beta$ può essere scritto come [3]:

$$\begin{cases} u_\alpha = Ri_\alpha + L_d \dfrac{di_\alpha}{dt} + e_\alpha \\ u_\beta = Ri_\beta + L_q \dfrac{di_\beta}{dt} + e_\beta \end{cases} \tag{5.1}$$

Dove

- u_α, u_β sono le tensioni dello statore;
- i_α, i_β sono le correnti dello statore;
- R è la resistenza dello statore;
- L_d, L_q sono le induttanze dello statore;
- e_α, e_β sono le Back-EMF.

si ha che le Back-EMF valgono:

$$e_\alpha = -Esin\theta_e, e_\beta = Ecos\theta_e \tag{5.2}$$

Dove:

- E è l'ampiezza delle Back-EMF, e soddisfa:

$$E = (L_d - L_q)\left(\omega_e i_d - \frac{di_q}{dt}\right) + \omega_e \psi_f \tag{5.3}$$

- θ_e è la posizione elettrica del rotore;
- ω_e è la velocità angolare elettrica;
- ψ_f è il flusso concatenato dei magneti permanenti.
- L_d, L_q sono le induttanze dello statore

5.1.1 Algoritmo Super-Twisting Sliding Mode

Come mostrato in [2], la formula che esprime l'algoritmo super-twisting è espressa come:

$$\begin{cases} \dot{\hat{x}}_1 = -k_1|\hat{x}_1 - x_1|\,\text{sign}(\hat{x}_1 - x_1) + \hat{x}_2 + \rho_1, \\ \dot{\hat{x}}_2 = -k_2\,\text{sign}(\hat{x}_1 - x_1) + \rho_2 \end{cases} \tag{5.4}$$

- x_i sono le variabili di stato,
- $\hat{x}_i$ è la stima delle variabili di stato,
- k_i sono i guadagni dell'osservatore sliding mode,
- $sign()$ è la funzione segno,
- ρ_i sono i termini di perturbazione

5.1.2 Condizioni di Stabilità dell'Algoritmo

In [5] sono state trovate le condizioni per la stabilità dell'algoritmo super-twisting: se ρ_1, ρ_2 soddsfano le seguenti condizioni la stabilità è garantita,

$$\rho_1 \leq \delta_1 |x_1|^{\frac{1}{2}}, \quad \rho_2 = 0$$

dove δ_1 è una costante positiva e i guadagni dell'osservatore soddisfano

$$k_1 > 2\delta_1, \quad k_2 > \frac{k_1 5\delta_1 k_1 + 4\delta_1^2}{12(k_1 - 2\delta_1)} \tag{5.5}$$

5.2 Stima delle Back-EMF

Come mostrato in [3] è possibile riscrivere il modello matematico dei motori PMSM:

$$\begin{cases} \frac{di_\alpha}{dt} = -\frac{R}{L} i_\alpha + \frac{u_\alpha}{L} - \frac{e_\alpha}{L} \\ \frac{di_\beta}{dt} = -\frac{R}{L} i_\beta + \frac{u_\beta}{L} - \frac{e_\alpha}{L} \\ e_\alpha = -\psi_f \omega_e \sin\theta_e \\ e_\beta = \psi_f \omega_e \cos\theta_e \end{cases} \tag{5.6}$$

dove:

- i_α, i_β sono le correnti di fase,
- u_α, u_β sono le tensioni di fase,
- e_α, e_β sono le Back-EMF,
- $\theta_e = p_n\theta$ è la posizione elettrica, p_n è il numero delle coppie polari,
- ψ_f è il flusso concatenato dei magneti permanenti.

Possiamo sostituire alle correnti di fase con la loro stima. Siano $\hat{i}_\alpha, \hat{i}_\beta$ le correnti stimate dall'osservatore:

$$\begin{cases} \frac{d\hat{i}_\alpha}{dt} = -\frac{R}{L}\hat{i}_\alpha + \frac{1}{L}u_\alpha + e_\alpha \\ \frac{d\hat{i}_\beta}{dt} = -\frac{R}{L}\hat{i}_\beta + \frac{1}{L}u_\beta + e_\beta \end{cases} \tag{5.7}$$

Definiamo $\tilde{i}_\alpha = \hat{i}_\alpha - i_\alpha$ e $\tilde{i}_\beta = \hat{i}_\beta - i_\beta$ gli errori di stima delle correnti.

Quando l'errore raggiunge la superficie di scorrimento, ovvero quando $\dot{\tilde{i}}_\alpha = \dot{\tilde{i}}_\beta = 0$, valgono le seguenti uguaglianze [9]:

$$\begin{bmatrix} \hat{E}_\alpha \\ \hat{E}_\beta \end{bmatrix} = \begin{bmatrix} k1|\tilde{i}_\alpha|^{1/2} sign(\tilde{i}_\alpha) + \int_0^t k2 sign(\tilde{i}_\alpha) dt \\ k1|\tilde{i}_\beta|^{1/2} sign(\tilde{i}_\beta) + \int_0^t k2 sign(\tilde{i}_\beta) dt \end{bmatrix} \tag{5.8}$$

- $\hat{E}_\alpha, \hat{E}_\beta$ sono le Back-EMF stimate,
- k_1, k_2 sono i guadagni dell'osservatore,
- $\tilde{i}_\alpha, \tilde{i}_\beta$ gli errori di stima delle correnti i_α, i_β

In questo progetto la funzione segno è stata sostituita con una funzione di saturazione per diminuire il fenomeno del chattering

$$Sat_k(\tilde{i}, \varepsilon) = \begin{cases} \frac{\tilde{i}_k}{\varepsilon} & \text{se } |\tilde{i}_k| < \varepsilon, \\ sign(\tilde{i}_k) & \text{se } |\tilde{i}_k| > \varepsilon. \end{cases} \tag{5.9}$$

5.3 Modello Matematico del PLL

Infine, è stato utilizzato un Phase-Locked Loop (PLL) per calcolare la posizione del rotore partendo dalla stima delle Back-EMF.
Definiamo l'errore di stima delle Back-EMF [4]:

$$\Delta E = -\hat{e_\alpha} cos(\hat{\theta}_e) - \hat{e_\beta} sin(\hat{\theta}_e) = \psi_f \omega_e sin(\Delta\theta_e) \tag{5.10}$$

con $\Delta\theta_e = \hat{\theta}_e - \theta_e$.
Quando il PLL ha individuato la posizione del rotore, $sin(\hat{\theta}_e)$ tende a θ_e, e la formula (4.9) può essere riscritta come:

$$\Delta E \approx -\psi_f \omega_e \Delta\theta_e \tag{5.11}$$

Il PLL è stato realizzato utilizzando un regolatore PI, per cui la sua funzione di trasferimento è [2]:

$$\frac{\hat{\theta}_e}{\theta_e} = \frac{k_p s + k_i}{s^2} \tag{5.12}$$

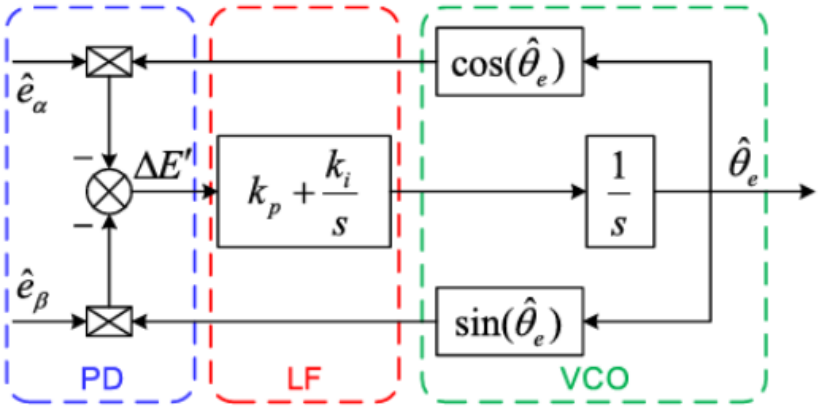

Figure 5.1: *Diagramma di un PLL*

Chapter 6

Osservatore - Implementazione

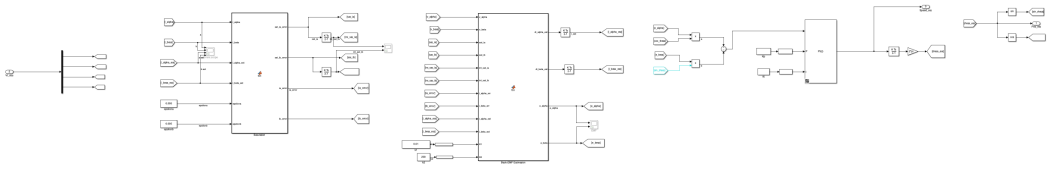

Figure 6.1: *Implementazione dell' Osservatore*

In questo capitolo verrà spiegato come è stato implementato l'osservatore utilizzando Simulink all'interno del progetto sviluppato da NXP.
L'osservatore è stato suddiviso in tre elementi: una funzione di saturazione, una funzione per stimare le Back-EMF, un PLL.

6.1 Input dell'Osservatore

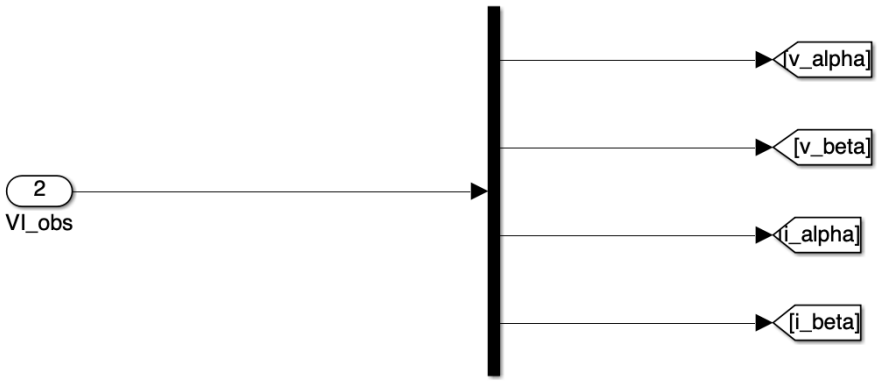

Figure 6.2: *Input*

In figura 6.2 è mostrato l'input dell'osservatore, come anticipato l'ingresso della funzione consiste nelle tensioni e correnti misurate, espresse sugli assi $\alpha - \beta$.

6.2 Funzione Saturazione

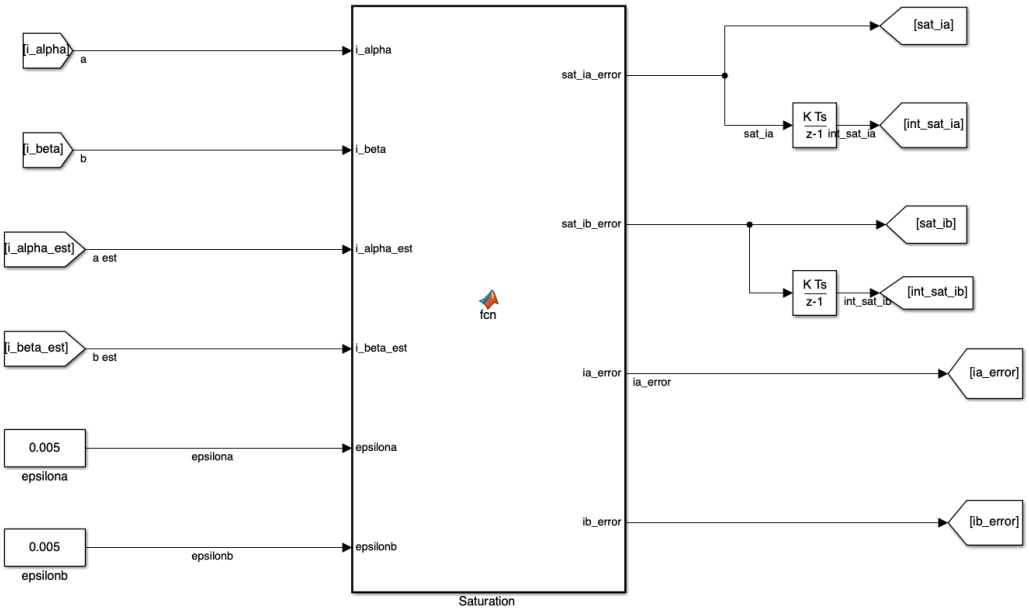

Figure 6.3: *Saturazione*

In figura 6.3 è mostrata la funzione di saturazione. L'input della funzione consiste nelle correnti i_α e i_β misurate, la loro stima, e due parametri ε, questi ultimi due parametri consentono di regolare la reattività rispetto all'errore, infatti la saturazione avrà modulo massimo solo se l'errore è maggiore della tolleranza epsilon, altrimenti la saturazione è proporzionale al modulo dell'errore. Lo scopo di questa funzione è quello di mitigare le oscillazioni causate dalla funzione sign(), permette infatti di diminuire il modulo quando l'errore è più piccolo dell'ε scelto. L'output della funzione consiste nella saturazione calcolata per entrambe le correnti, gli integrali della saturazione e l'errore di stima. Tutti questi dati sono necessari per stimare le forze elettromotrici indotte.

6.2.1 Implementazione Matlab

```
1 function [sat_ia_error, sat_ib_error, ia_error, ib_error]
      = fcn( i_alpha, i_beta, i_alpha_est, i_beta_est,
    epsilona, epsilonb)
```

```
2
3  epsilon_a = epsilona;
4  epsilon_b = epsilonb;
5
6
7  ia_error = i_alpha_est - i_alpha;
8  ib_error = i_beta_est - i_beta;
9
10 if abs(ia_error) > epsilon_a
11     sat_ia_error = sign(ia_error);
12 else
13     sat_ia_error = abs(ia_error) / epsilon_a;
14 end
15
16 if abs(ib_error) > epsilon_b
17     sat_ib_error = sign(ib_error);
18 else
19     sat_ib_error = abs(ib_error) / epsilon_b;
20 end
```

6.3 Stima delle Back-EMF

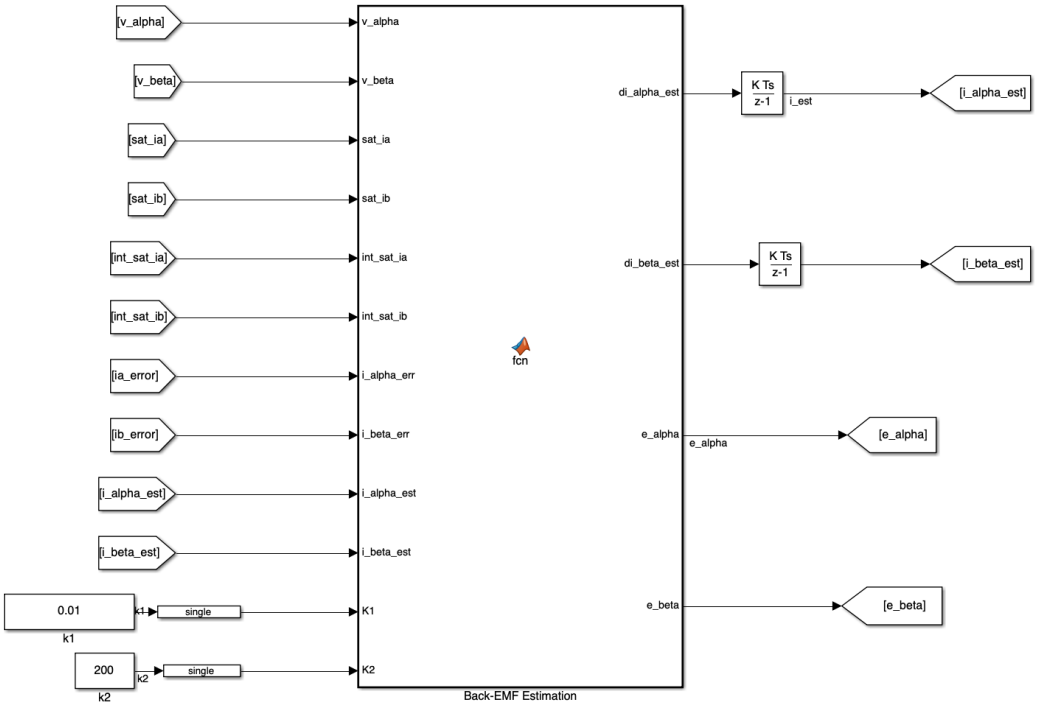

Figure 6.4: *Stima delle Back-EMF*

Il prossimo passo è quello di stimare le Back-EMF.
le Back-EMF sono generate dalla rotazione del rotore in presenza di un campo magnetico e sono direttamente correlate alla velocità del motore e alla posizione del rotore. Per questo motivo, la loro stima accurata consente di determinare indirettamente le grandezze meccaniche del sistema, eliminando la necessità di sensori fisici.

Come detto nel capitolo procedente, la stima delle Back-EMF è realizzata a partire dalle equazioni differenziali che descrivono il comportamento elettrico del motore nel riferimento $\alpha - \beta$. La precisione della stima delle Back-EMF è essenziale per garantire prestazioni ottimali del controllo, poiché errori significativi si riflettono direttamente nella qualità delle stime di velocità e posizione.
In figura 6.4 è mostrato il blocco simulink che implementa la funzione. Questa ha in input:

- Le tensioni dello statore
- Le saturazioni calcolate dalla precedente funzione
- Gli integrali delle saturazioni

- L'errore di stima per entrambe le correnti
- Il valore della precedente stima delle correnti
- I due guadagni K1 e K2

Grazie alla funzione di saturazione abbiamo tutti i dati necessari per stimare le Back-EMF. Tramite queste è inoltre possibile ricavare le derivate delle correnti $\hat{i}_\alpha$ e $\hat{i}_\beta$ e, applicando un integrale, si ottiene dunque una nuova stima delle due correnti.

6.3.1 Implementazione Matlab

```
function [di_alpha_est, di_beta_est, e_alpha, e_beta] =
    fcn(v_alpha, v_beta, sat_ia, sat_ib, int_sat_ia,
   int_sat_ib, i_alpha_err, i_beta_err, i_alpha_est,
   i_beta_est, K1, K2)

k1 = K1;
k2 = K2;
R = 0.6367; %pmsm.Rs;
L = 0.000375; %pmsm.Ld;

z_alpha = k1*sqrt(abs(i_alpha_err))*sat_ia + k2*
   int_sat_ia;
z_beta = k1*sqrt(abs(i_beta_err))*sat_ib + k2*int_sat_ib;

di_alpha_est = -(R/L)*i_alpha_est + (1/L)*v_alpha - (1/L)
   *z_alpha;
di_beta_est = -(R/L)*i_beta_est + (1/L)*v_beta - (1/L)*
   z_beta;

e_alpha = z_alpha;
e_beta = z_beta;
```

6.4 Phase-Locked Loop

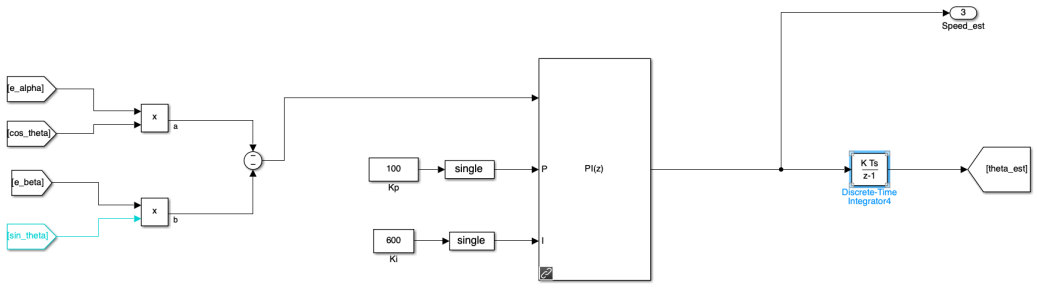

Figure 6.5: *PLL*

Il PLL riceve in input le Back-EMF appena calcolate e utilizza un controllore PI per ridurre l'errore di stima. Con questa funzione possiamo finalmente ottenere una stima della velocità del motore e dunque della posizione del rotore.

6.5 Osservatore inserito nel sistema di controllo delle correnti

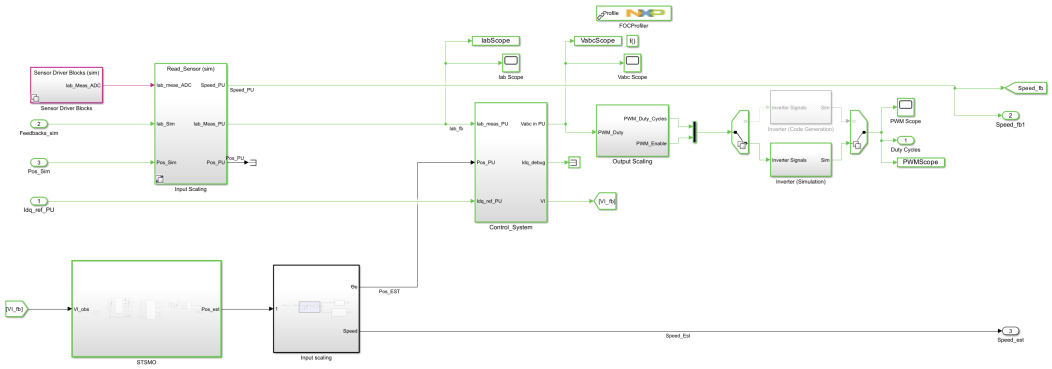

Figure 6.6: *Osservatore inserito nel sistema di controllo delle correnti*

6.5.1 Scaling

È stato utilizzato un sistema di scaling per rendere compatibile la posizione stimata dall'osservatore con il sistema di controllo utilizzato da NXP. Con questa funzione l'angolo, inizialmente espresso in radianti, viene normalizzato tra [0,1]. Successivamente vengono calcolate la posizione elettrica e la velocità.

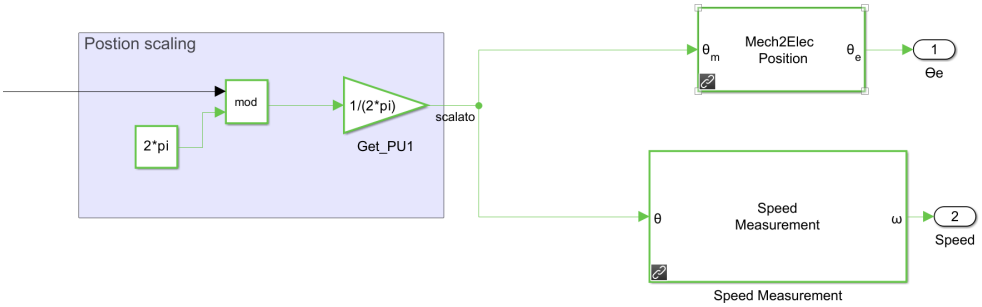

Figure 6.7: *Scaling della posizione*

Chapter 7

Simulazione

La simulazione rappresenta una fase fondamentale nello sviluppo dell'osservatore, poiché consente di validare il funzionamento degli algoritmi in un ambiente controllato prima di passare all'implementazione su hardware reale. Attraverso l'analisi dei risultati ottenuti, è possibile valutare la capacità dell'osservatore di stimare correttamente la velocità e altre grandezze del sistema, identificando eventuali criticità o margini di miglioramento.

Un aspetto cruciale evidenziato durante questa fase è l'importanza del corretto tuning dei parametri dell'osservatore. Una configurazione non ottimale può compromettere la precisione delle stime o influire negativamente sulla stabilità del sistema. Pertanto, i test in simulazione sono stati utilizzati non solo per verificare il comportamento complessivo, ma anche per calibrare i parametri chiave dell'osservatore in modo iterativo.

In questo capitolo verranno presentati i risultati ottenuti nelle varie simulazioni, evidenziando il processo di tuning e il suo impatto sulle prestazioni dell'osservatore.
In simulazione sono state testate le prestazioni dell'osservatore anche variando il riferimento della velocità del motore, in modo da mettere in evidenza eventuali comportamenti indesiderati.

7.1 Tuning dell'Osservatore

Una parte fondamentale di questo progetto è stata l'attività di tuning, ovvero variare i parametri dell'osservatore cercando di ottenere una stima della posizione e della velocità più vicina possibile rispetto ai valori della simulazione. Come detto in precedenza l'osservatore mette a disposizione sei parametri:

- $\varepsilon_\alpha, \varepsilon_\beta$ per limitare gli errori delle correnti
- k_1, k_2 i guadagni dell'osservatore
- k_p, k_i guadagni del regolatore PI

È necessario evidenziare l'importanza di assegnare i giusti valori a questi parametri per ottenere una stima valida dello stato del motore.
Di seguito sono riportati i risultati di alcuni dei test svolti durante la fase di tuning. In questi test si sono verificate le prestazioni dell'osservatore rispetto a due tipi di riferimento della velocità, rampa e gradino.

7.2 Test Iniziali

7.2.1 Ingresso a Gradino

In questi test andiamo a valutare la risposta dell'osservatore rispetto ad un riferimento a gradino.
In un primo momento l'osservatore è stato posto al di fuori della logica di controllo, in modo che si potesse testare la qualità della stima dell'osservatore.
Dopo numerose simulazioni sono stati individuati questi valori dei parametri:

- $\varepsilon_\alpha = \varepsilon_\beta = 0.0001$
- k1 = 0.5
- k2 = 200
- kp = 150
- ki = 100

In seguito sono presentati i grafici delle grandezze di interesse, ottenuti con durante la simulazioni.
In figura 7.1 è riportato il grafico che confronta la velocità stimata e la velocità simulata del motore

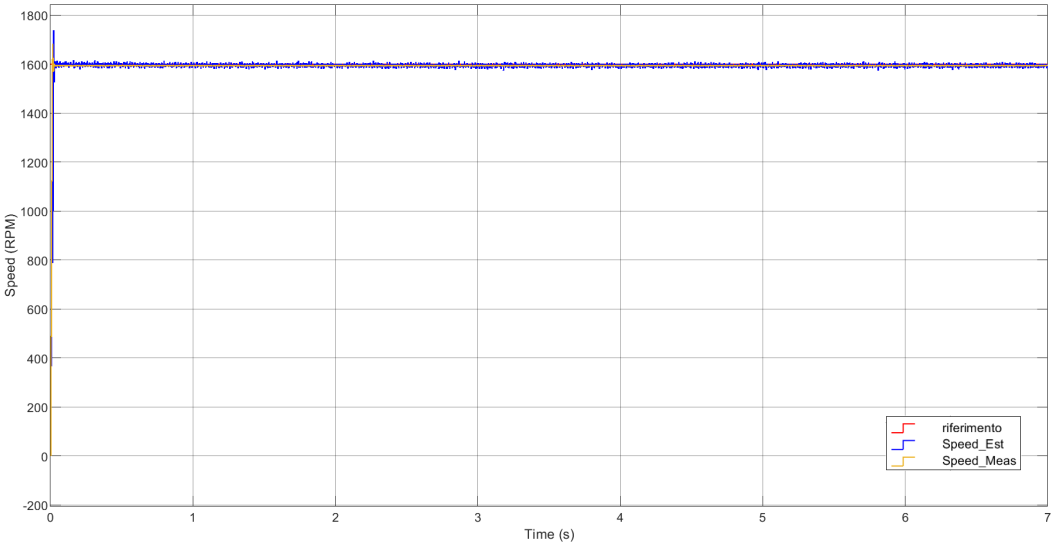

Figure 7.1: *Velocità Stimata*

in 7.2 l'errore di stima della velocità espresso in percentuale,

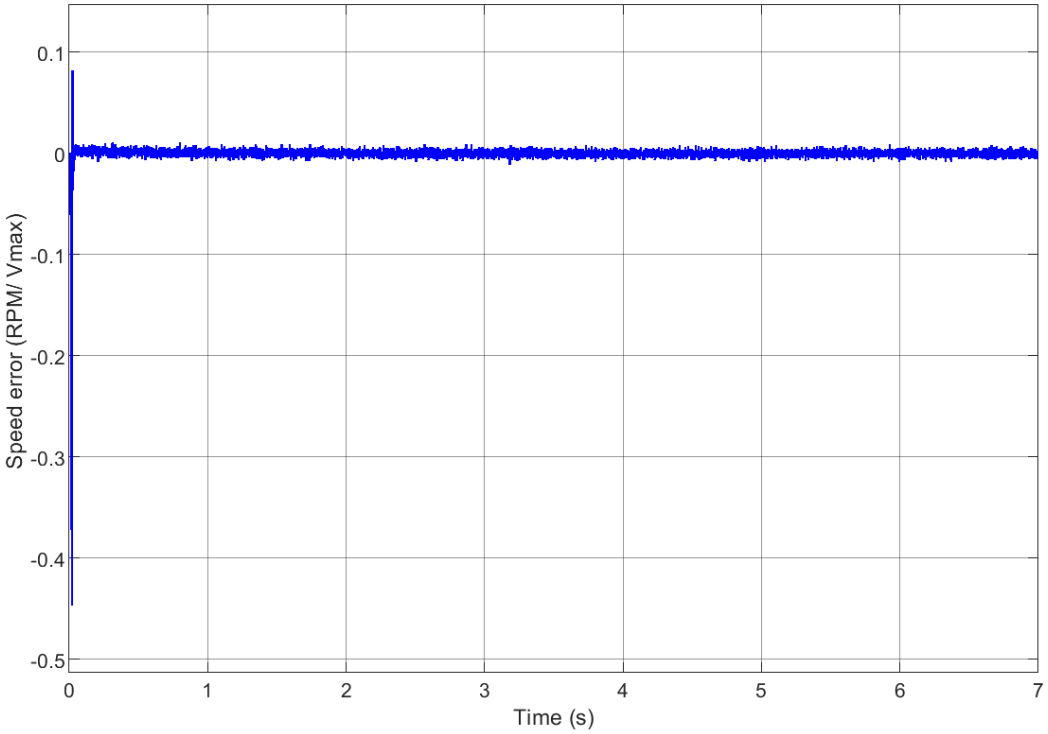

Figure 7.2: *Errore di stima della velocità*

di seguito nei grafici 7.3 e 7.4, il confronto tra la posizione stimata e quella simulata, e l'errore di stima della posizione.

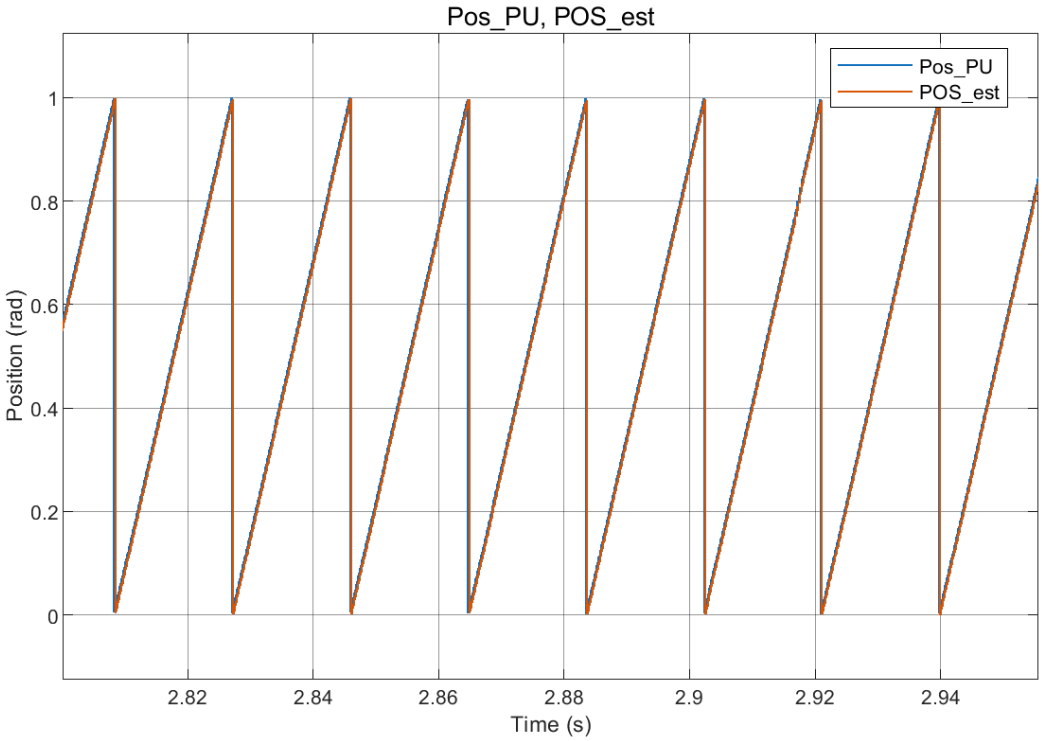

Figure 7.3: *Posizione Stimata*

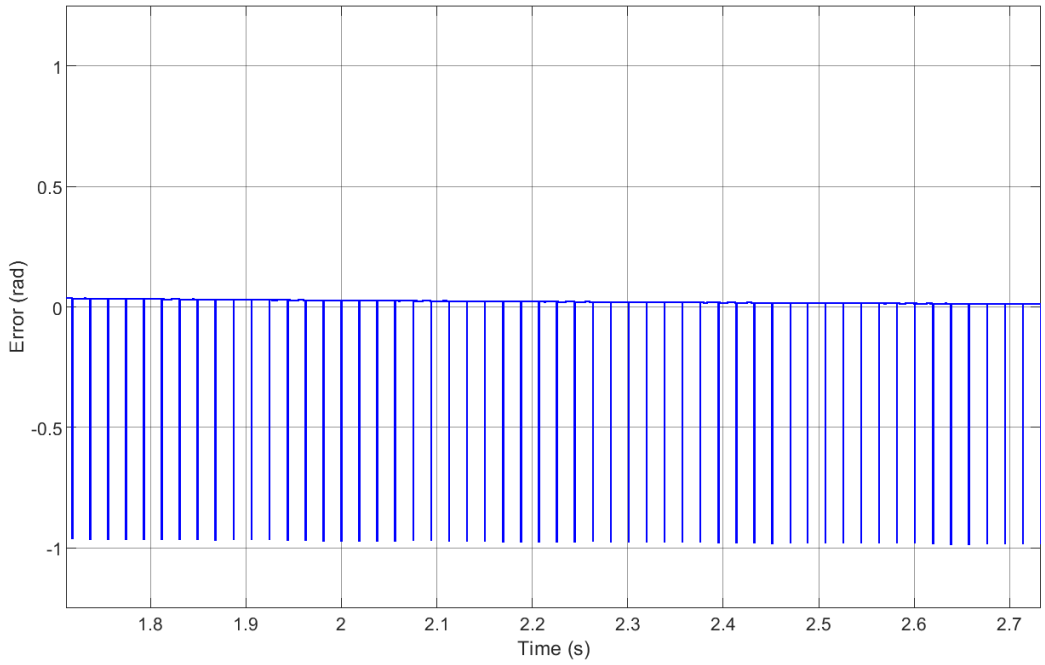

Figure 7.4: *Errore di stima della posizione*

nel grafico 7.5 è riportata la stima delle back-EMF per entrambi gli assi del riferimento $\alpha - \beta$

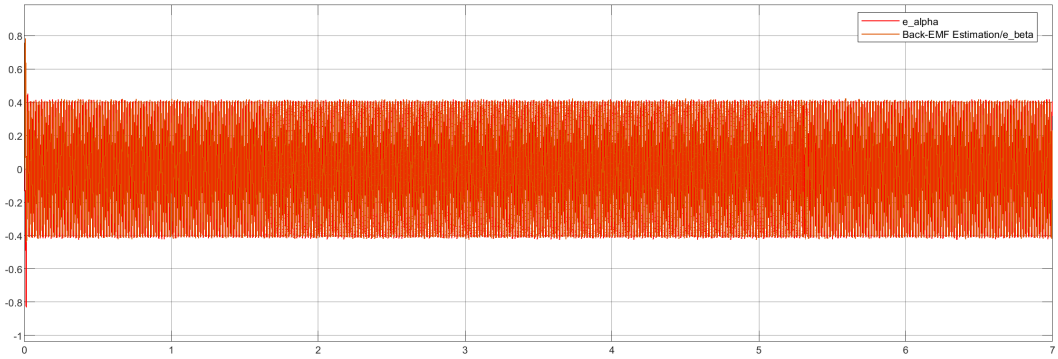

Figure 7.5: *Back-EMF Stimate*

Infine in 7.6 l'errore di stima delle correnti i_α, i_β

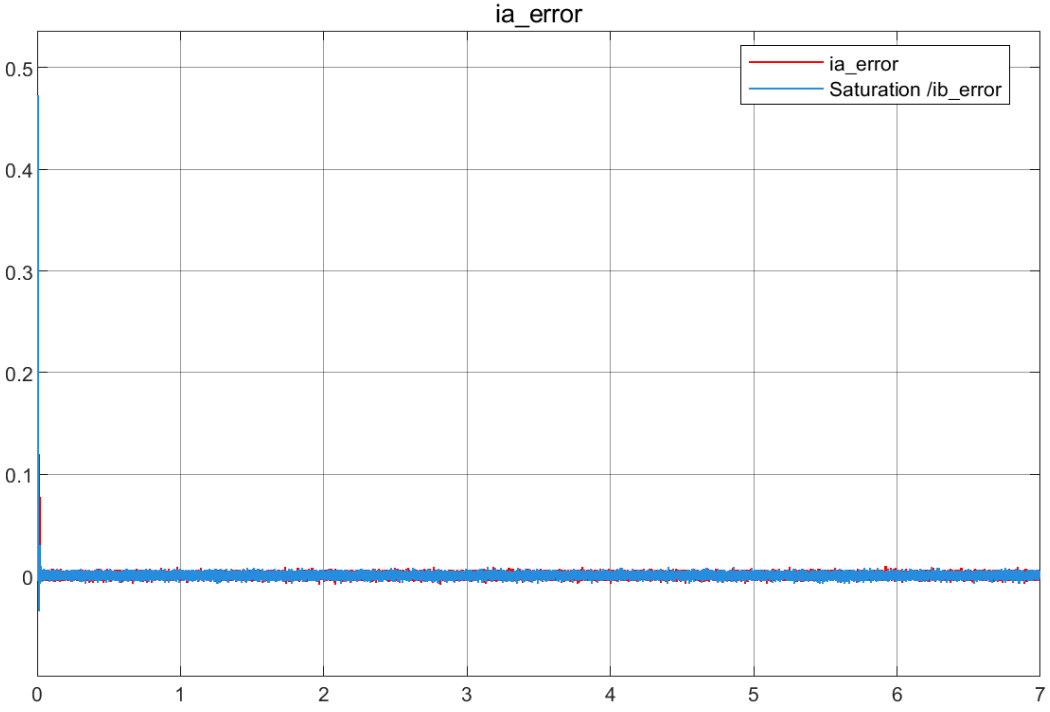

Figure 7.6: *Errore di Stima delle Correnti*

7.2.2 Discussione dei risultati

Come si può osservare dai grafici ottenuti durante queste simulazioni, l'osservatore riesce a stimare correttamente gli stati del motore anche a seguito di cambiamenti improvvisi, come l'input a gradino, tuttavia è ancora presente il fenomeno del chattering. La causa di questo fenomeno è correlata a una cattiva stima delle Back-EMF, che implica la necessità di individuare nuovi parametri che consentono all'osservatore di migliorare la qualità delle stime. Per valutare

la qualità dell'osservatore sono stati calcolati gli indici prestazionali ISE, IAE, ITAE:

- ISE = 0.0083
- IAE = 0.029
- ITAE = 0.033

Questi valori suggeriscono che l'osservatore ha avuto una buona performance, e ha stimato la velocità del motore con errori totali relativamente bassi.

7.2.3 Ingresso a Rampa

In seguito sono presentati i risultati della simulazione in cui la velocità di riferimento è una funzione rampa. L'obiettivo di questa simulazione è quello di valutare la robustezza del risultato anche con variazioni più lente. Come nella sezione precedente sono riportati i grafici:

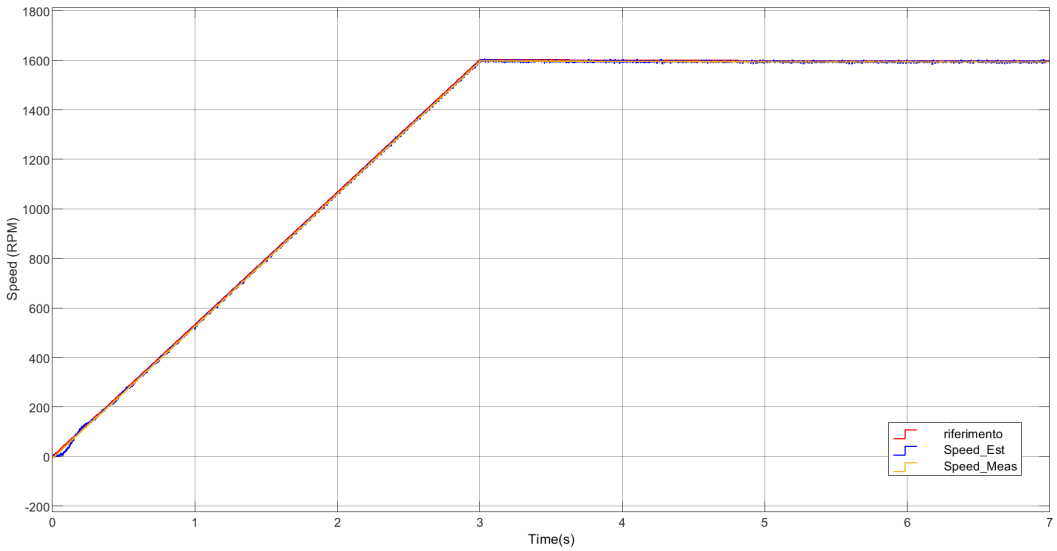

Figure 7.7: *Velocità Stimata*

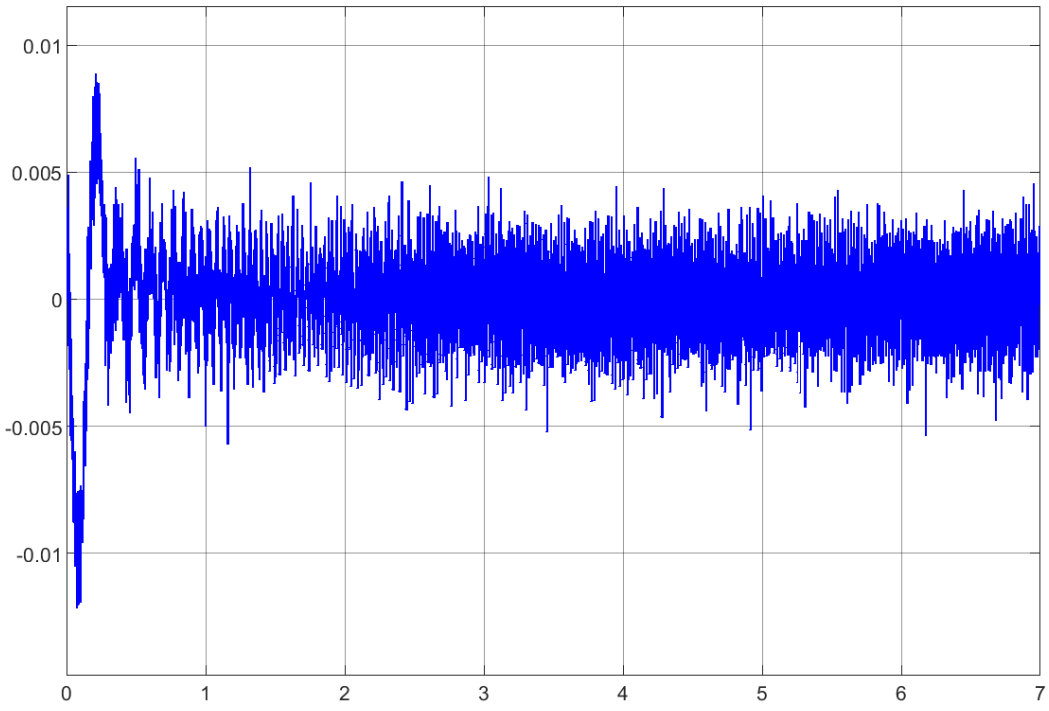

Figure 7.8: *Errore di stima della Velocità(%)*

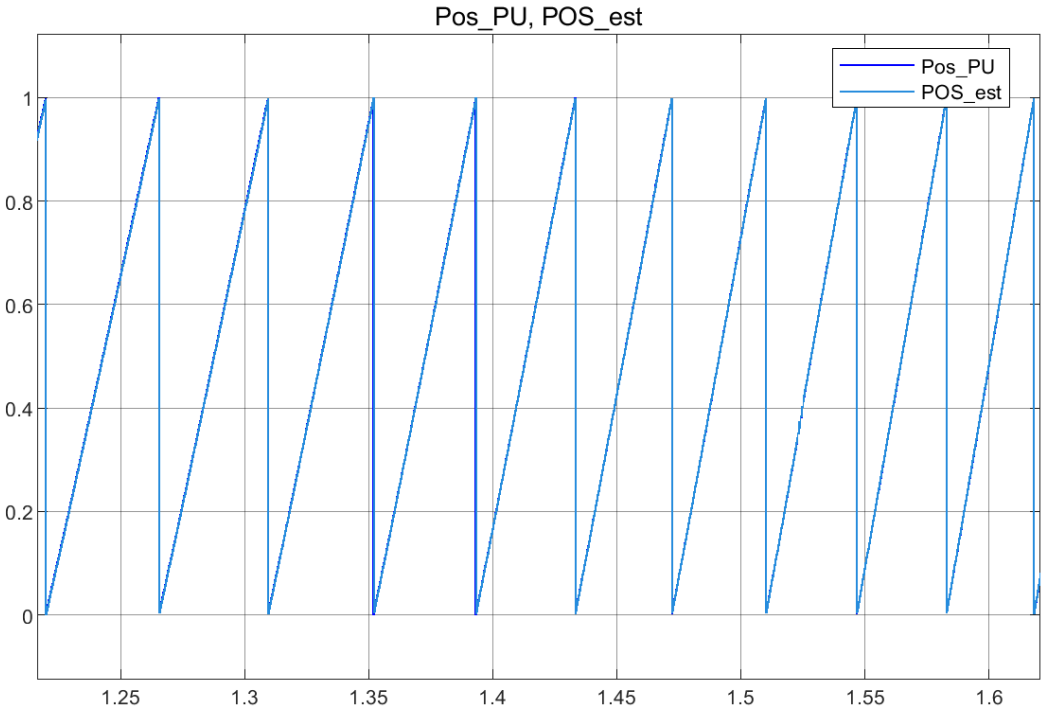

Figure 7.9: *Stima della Posizione*

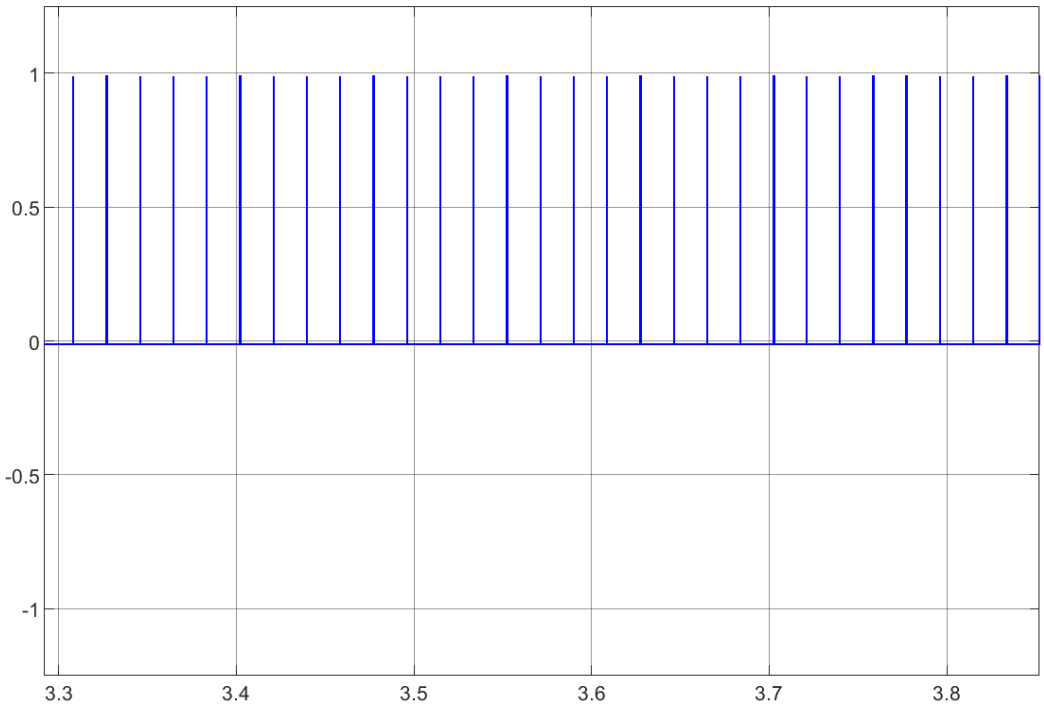

Figure 7.10: *Errore di stima della posizione*

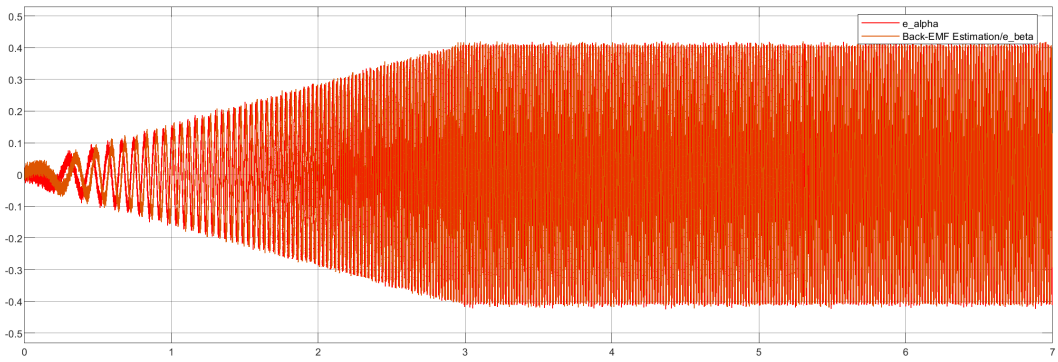

Figure 7.11: *Back-EMF stimate*

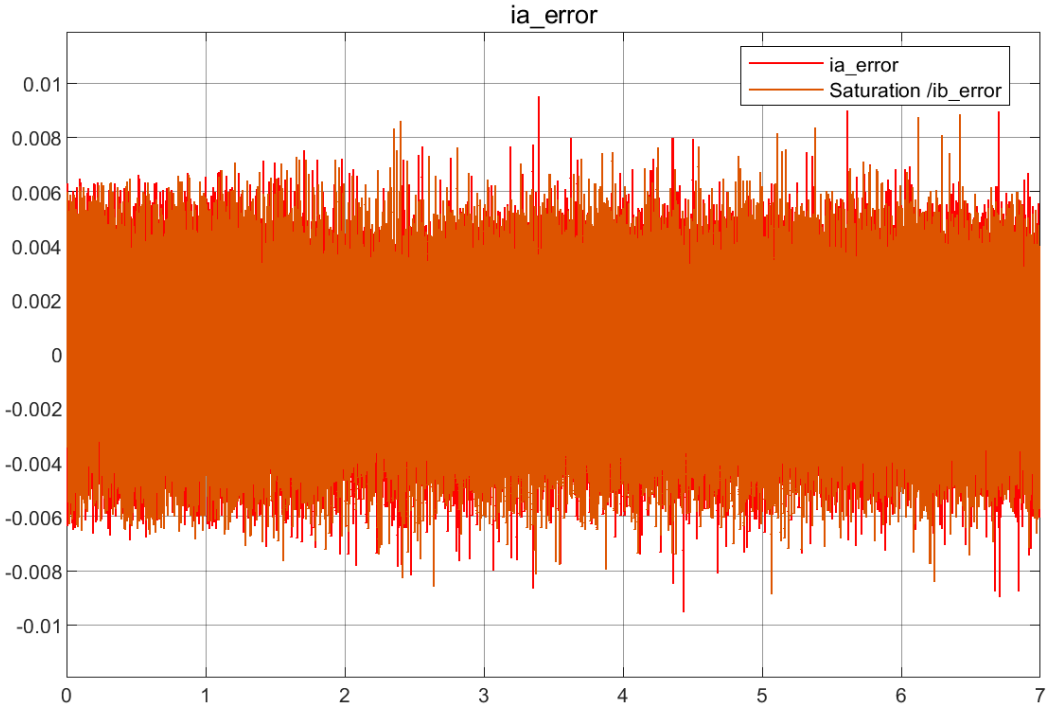

Figure 7.12: *Errore di stima della correnti*

7.2.4 Discussione dei risultati

Come si evince dai precedenti grafici anche in questo caso l'osservatore riesce a produrre una buona stima delle variabili di interesse.
Nelle figura 7.7 si può notare che la velocità stimata parte con un leggero ritardo rispetto alla velocità simulata, è importante ricordare che la velocità stimata dipende dalla stima delle forze elettromotrici indotte, le quali sono proporzionali alla velocità del rotore. Di conseguenza, a velocità molto basse, le FEM diventano deboli, riducendo l'accuratezza della stima e compromettendo la capacità del PLL di tracciare correttamente la posizione del rotore.

7.3 Osservatore Inserito nella Logica di Controllo

In questa fase sono state effettuate delle simulazioni per verificare il comportamento del motore quando la velocità e la posizione vengono stimate dall'osservatore e non sono provenienti dai sensori.

Dopo aver analizzato il comportamento dell'osservatore in un contesto isolato dalla logica di controllo, è stato effettuato un ulteriore passo verso la validazione del sistema. Nelle simulazioni successive, l'osservatore è stato integrato nella logica di controllo del motore, con l'obiettivo di verificare se le

stime fornite fossero sufficientemente accurate e stabili da supportare il controllo della velocità e delle correnti.
Tuttavia, già dalle prime simulazioni sono emersi problemi significativi, il tuning iniziale si è rivelato insufficiente quando l'osservatore è stato accoppiato alla logica di controllo. Di conseguenza, è stato necessario rivedere sia i parametri caratteristici dell'osservatore stesso, sia quelli relativi al controllo.
Al termine di questa fase sono stati individuati i seguenti parametri:

- $\varepsilon_\alpha = 0.0001$
- $\varepsilon_\beta = 0.0001$
- $k_1 = 0.5$
- $k_2 = 200$
- $k_p = 70$
- $k_i = 2500$

Di seguito sono riportati i risultati delle simulazioni.

7.3.1 Riferimento a gradino

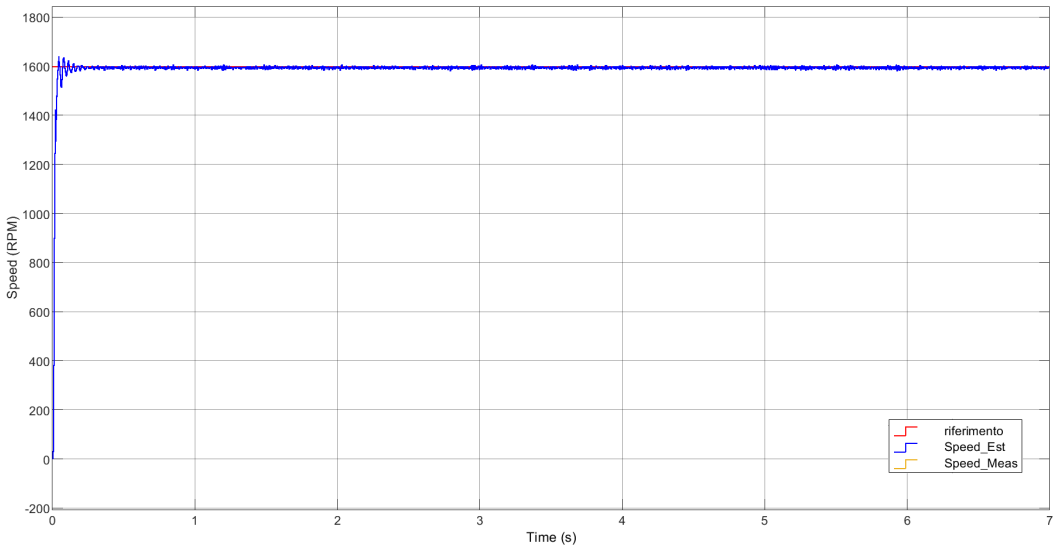

Figure 7.13: *Velocità del motore*

Come si può osservare in figura 7.13, il sistema riesce a controllare anche utilizzando le stime dell'osservatore, anche se la risposta presenta ancora delle oscillazioni indesiderate, lasciando un margine di miglioramento. Per questa simulazione gli indici prestazionali valgono:

- ISE: 0.0066
- IAE: 0.0310
- ITAE: 0.0662

Anche in questo caso gli indici hanno valori contenuti, confermando che il motore riesce a seguire correttamente il riferimento, anche utilizzando le grandezze stimate dall'osservatore.

7.3.2 Riferimento a rampa

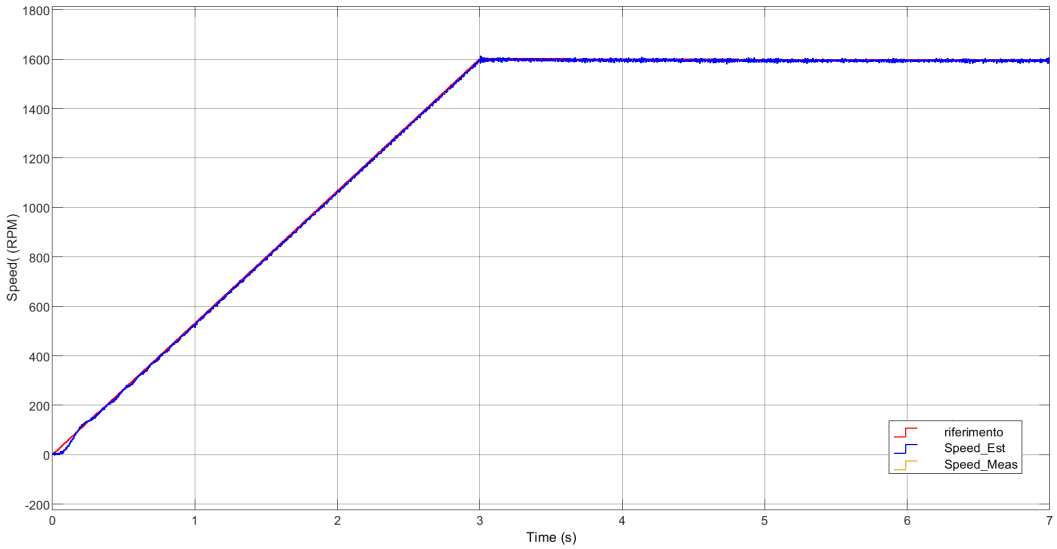

Figure 7.14: *Velocità del motore*

Anche in questa simulazione il motore è riuscito a seguire il riferimento senza l'utilizzo di sensori.

7.4 Discussione dei risultati

Attraverso le simulazioni è stato possibile verificare il comportamento dell'osservatore e la qualità delle sue stime. Inoltre è stato possibile confermare il funzionamento del motore anche senza l'utilizzo dei sensori.
Anche se ancora soggetto al fenomeno del chattering, il motore riesce a seguire la velocità di riferimento desiderata, sia con variazioni lente, come visto con il riferimento a rampa, sia con variazioni più rapide, come visto con il gradino.
Nella fase successiva verrà testato il codice sviluppato nel modello reale del motore, per validare i risultati ottenuti.

Chapter 8

Validazione

In questo capitolo, il codice sviluppato in Simulink è stato validato utilizzando il kit S32K144-EVB. Grazie al toolbox fornito da NXP, è stato possibile interfacciarsi con la scheda e caricare direttamente il codice sul dispositivo. Questo processo ha consentito di confrontare i risultati delle simulazioni con quelli ottenuti in condizioni reali.

Il grafico in figura 8.1 rappresenta la velocità del motore, indicata in viola, rispetto al riferimento, indicato in verde.

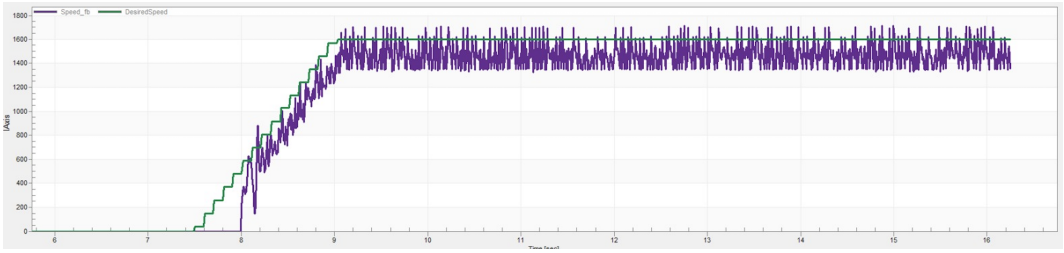

Figure 8.1: *Velocità del motore*

I risultati ottenuti dimostrano che l'osservatore sviluppato è in grado di stimare correttamente la velocità del motore e di seguirne il riferimento. Tuttavia, si notano oscillazioni e un errore residuo, fenomeni già presenti prima dell'implementazione dell'osservatore. Di conseguenza, le cause di tali oscillazioni non possono essere attribuite esclusivamente all'osservatore.

Chapter 9

Conclusioni e Sviluppi futuri

In questa tesi si è trattato dello sviluppo di un osservatore di stato, con l'obiettivo di sviluppare un controllo privo di sensori per un motore Brushless a magneti permanenti (PMSM). È stata sviluppata una soluzione basata sull'approccio sliding mode, implementando uno Super-Twisting Sliding Mode Observer (STSMO). L'obiettivo principale era fornire una stima affidabile della velocità del rotore, riducendo la dipendenza da sensori fisici e migliorando la robustezza del sistema.

Dopo un'analisi approfondita della teoria alla base dell'algoritmo Super-Twisting e della logica utilizzata per il progetto, l'osservatore è stato testato attraverso simulazioni e successivamente validato su hardware reale utilizzando il kit SM-CSPTE1AK144. I risultati ottenuti hanno dimostrato che l'osservatore è in grado di stimare la velocità e di seguire il riferimento desiderato, nonostante la presenza di oscillazioni e di un errore residuo.

Sono proposti come possibili sviluppi futuri:

- Perfezionare l'osservatore, individuando dei parametri che ottimizzino la qualità delle stime
- Utilizzare funzioni alternative alla saturazione, per smorzare ulteriormente le oscillazioni
- Integrare un controllo migliorato, ampliando l'approccio sliding mode anche al controllo delle correnti.

Bibliography

[1] Eleonora Brasili. Controllo sensorless basato su un osservatore sliding mode per un pmsm: validazione sperimentale su un elettrodomestico, 2024.

[2] Shuo Chen, Xiao Zhang, Xiang Wu, Guojun Tan, and Xianchao Chen. Sensorless control for ipmsm based on adaptive super-twisting sliding- mode observer and improved phase-locked loop. 2019.

[3] Wang Dazhi, Liu Jingjing, Miao Shu, Yuan Tianqing, Li Ye, Tian Wenbo, and Liu Zhen. Rotor position estimation method for permanent magnet synchronous motor based on super-twisting sliding mode observer. In *2018 37th Chinese Control Conference (CCC)*, pages 5634–5638, 2018.

[4] Liu Gang, Zhang Haifeng, and Song Xinda. Position-estimation deviation-suppression technology of pmsm combining phase self-compensation smo and feed-forward pll. *IEEE Journal of Emerging and Selected Topics in Power Electronics*, 9(1):335–344, 2021.

[5] Jaime A. Moreno and Marisol Osorio. A lyapunov approach to second-order sliding mode controllers and observers. *IEEE Conference on Decision and Control*, 2008.

[6] Giuseppe Orlando. Tecniche di controllo a struttura variabile per un veicolo sottomarino comandato a distanza, 1996.

[7] R. H. Park. Two-reaction theory of synchronous machines generalized method of analysis-part i. *Transactions of the American Institute of Electrical Engineers*, 48(3):716–727, 1929.

[8] Muhammad Rafiq, Saeed ur Rehman, Fazal ur Rehman, Qarab Raza Butt, and Irfan Awan. A second order sliding mode control design of a switched reluctance motor using super twisting algorithm. *Simulation Modelling Practice and Theory*, 25:106–117, 2012.

[9] Qiwei Xu, Donghao Jiang, Yiming Wang, Xuefeng Zhang, Jincheng Liu, and Yangming Chen. Variable-step close-loop angle compensation method of pmsm rotor position estimation based on super-twisting sliding-mode observer using tangent reaching law. *Energy Reports*, 9:356–361, 2023.

The 8th International Conference on Sustainable and Renewable Energy Engineering.