



UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea Triennale in Ingegneria

Informatica e dall'Automazione

**STUDIO E SVILUPPO DI APPLICAZIONI DI CONTROLLO E
ACQUISIZIONE DATI IN TEMPO REALE PER DSP**

**STUDY AND DEVELOPMENT OF REAL-TIME DATA CONTROL AND
ACQUISITION APPLICATIONS FOR DSP**

Relatore: Chiar.mo

Prof. **Ippoliti Gianluca**

Correlatore: Chiar.mo

Prof. **Orlando Giuseppe**

Tesi di Laurea di:

Luca Ranucci

A.A. 2022 / 2023

Indice

Introduzione	4
Capitolo 1: Generalità sui DSP, hardware e software usati	6
1.1 Struttura e funzionamento di un'applicazione DSP	6
1.2 Modalità di funzionamento dei DSP	8
1.3 Hardware usato	10
1.3.1 Motore elettrico con dinamo tachimetrica	10
1.3.2 Amplificatore a mosfet	11
1.3.3 Partitore di tensione con condensatore	12
1.3.4 Scheda MicroDAQE200 E2000 AC05-DAC05-12	13
1.4 Struttura completa sistema	16
1.5 Software usato	17
1.6 Applicazioni standalone ed external-mode	19
1.7 Blocchi XCos.....	20
Capitolo 2: Identificazione	26
2.1 Identificazione del processo da controllare	26
2.2 Risposta in frequenza	29
2.2.1 Dati raccolti	32
2.3 Risposta Indiciale.....	38
Capitolo 3: Progettazione controllore	42
3.1 Metodi e specifiche	42
3.2 Sintesi in frequenza	42
3.2.1 Simulazione	46
3.3 Sintesi con il luogo delle radici	48
3.3.1 Simulazione	52
Capitolo 4: Controllo	54
4.1 Controllo del motore elettrico	54
4.2 Controllore progettato in frequenza.....	55
4.3 Controllore progettato col luogo delle radici.....	59
Conclusioni	64
Bibliografia e sitografia	65

Introduzione

L'avvento dei DSP, dall'inglese *digital signal processor*, ha rivoluzionato molte industrie portando l'elaborazione di segnali ad un nuovo livello. Questi microprocessori svolgono un ruolo fondamentale in molti settori, spaziando dalle telecomunicazioni, ai dispositivi medici, alla manipolazione di immagini, video e audio, e tanti altri.

In ogni ambito di interesse è stata sviluppata in modo approfondito un'ottimizzazione e un adattamento della tecnologia DSP, con specifici algoritmi, formule matematiche e tecniche implementative.

In generale il DSP è un microprocessore dedicato a processare segnali continui, anche in tempo reale, usando algoritmi specifici e complessi ad alta velocità. Ha infatti un'architettura ottimizzata a svolgere le operazioni matematiche più ricorrenti nei calcoli DSP con tempi ed efficienza energetica migliori di un processore multipurpose.

Questa tesi affronta lo studio e lo sviluppo di un'applicazione di controllo e acquisizione dati in tempo reale per DSP. Lo scopo è arrivare a controllare in retroazione un motore elettrico a corrente continua in velocità con il dispositivo MicroDAQ E2000, una scheda DSP finalizzata all'acquisizione alla manipolazione di dati in real-time.

Questo elaborato può essere diviso in quattro macro-parti:

- **Capitolo 1:** *Generalità sui DSP, hardware e software usati*
- **Capitolo 2:** *Identificazione del processo da controllare*

Viene definito qual è il sistema che si vuole identificare, si effettua una stima del tipo di modello matematico e poi si procede all'identificazione del sistema raccogliendo i dati necessari con la scheda MicroDAQ.

- **Capitolo 3:** *Progettazione del controllore*

Viene progettato un controllore per controllare il sistema in retroazione, tale progettazione è effettuata in due modi, tramite la sintesi in frequenza e la sintesi con il luogo delle radici, successivamente viene simulato il sistema in catena chiusa e testato il controllore.

- **Capitolo 4:** *Controllo del sistema*

Il motore elettrico viene controllato in retroazione in real-time con la scheda MicroDAQ e il controllore precedentemente sintetizzato.

Capitolo 1

Generalità sui DSP, hardware e software usati

1.1 Struttura e funzionamento di un'applicazione DSP

La struttura tipo di un'applicazione di digital signal processing è la seguente:

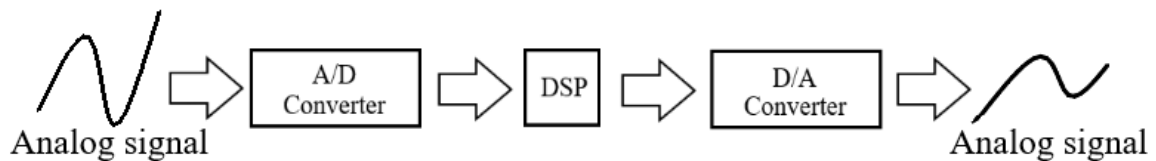


Figura 1: Struttura di un'applicazione DSP.

Input Analogico: Il segnale analogico viene dato in ingresso ad un ADC, ovvero un convertitore analogico-digitale, e in base al tempo di campionamento selezionato il convertitore manda interrupt al processore DSP e rende disponibile il campione digitale.

Processamento del segnale digitale: I dati in arrivo sono gestiti dagli algoritmi del DSP, quando il processore ha completato i calcoli richiesti i dati vengono mandati al DAC. Ogni campione digitale deve essere processato prima che arrivi l'interrupt successivo.

Output Analogico: Il DAC, convertitore digitale-analogico, converte i dati in uscita dal processore DSP in analogici nel successivo istante di

campionamento, filtrato opportunamente il segnale analogico viene poi ricostruito.

Ci sono tuttavia delle applicazioni dove alcune componenti non sono necessarie, per esempio il DAC per l'acquisizione dati o l'ADC per l'elaborazione di dati digitali.

Le componenti principali di un DSP sono:

- Program Memory: la memoria che contiene gli algoritmi che il DSP usa per l'elaborazione dei segnali.
- Data Memory: la memoria che contiene tutti i dati da analizzare e processare, ottimizzata per essere molto veloce.
- CPU: il processore che svolge tutte le operazioni matematiche necessarie per l'elaborazione dei dati e ha accesso alla Program Memory e alla Data Memory. È composto dal *moltiplicatore/accumulatore* (MAC), dall'*unità aritmetica* (ALU) e da un *barrel shifter*.
- Dispositivi di input/output: periferiche necessarie al DSP per interfacciarsi con gli ingressi e le uscite.

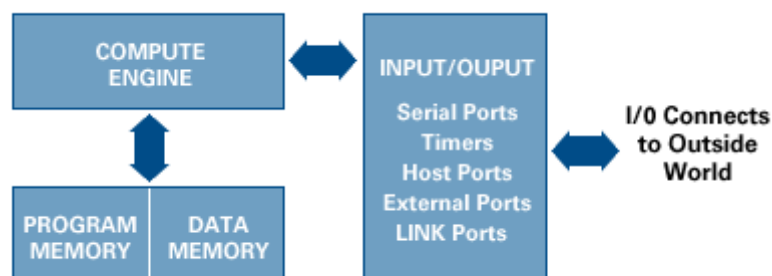


Figura 2: Componenti di un DSP.

1.2 Modalità di funzionamento dei DSP

I DSP operano principalmente in due modalità: offline e real-time.

Nella modalità offline tutti i dati da elaborare sono simultaneamente accessibili dal DSP, non è necessaria una manipolazione immediata dei dati e di solito questa avviene in un momento successivo, tipicamente con il vantaggio di avere a disposizione più risorse computazionali e potenzialmente più tempo a disposizione rispetto al tempo necessario per le applicazioni real-time, possono perciò essere usati algoritmi più complessi. La modalità off-line viene usata per esempio nei personal computer, nella pulizia di segnali audio e immagini, analisi di dati o apparecchiature scientifiche.

Nella modalità real-time l'elaborazione dei dati viene svolta nello stesso momento in cui viene acquisito o generato il segnale da elaborare, ciò è fondamentale per tutte quelle applicazioni dove si controlla un sistema in tempo reale e si necessita di una bassa latenza. Un'applicazione real-time DSP è di prima importanza nell'ambito dell'industria dell'automazione e della robotica dove si necessita di processare dati registrati dai sensori in tempo reale e aggiustare il segnale di controllo in maniera adeguata.

La modalità real-time necessita di un hardware specializzato al fine di ottenere tempi di esecuzione molto veloci e ritardi minimi nel processamento dei segnali.

Real-time DSP sono usati per esempio nelle manipolazioni live di audio e video, nei dispositivi medici, nei sistemi di comunicazioni come reti cellulari, o nei sistemi di guida autonoma.

1.3 Hardware usato

1.3.1 Motore elettrico con dinamo tachimetrica

Viene usato il Motore Selema Serie 30 con annessa dinamo tachimetrica.

<i>Motore Selema Serie 30</i>		<i>Dinamo tachimetrica</i>	
<i>Magneti</i>	in ferrite	<i>Magneti</i>	permanenti
<i>Coppia</i>	0.03 ÷ 2 Nm	<i>Massima velocità</i>	6000 rpm
<i>Grado di protezione</i>	IP44	<i>Grado di protezione</i>	IP44
<i>Classe di isolamento</i>	F	<i>Isolamento</i>	F
<i>Temperatura ambiente</i>	0 ÷ +40°C	<i>Temperatura di funzionamento</i>	0 ÷ +40°C
<i>Servizio</i>	continuo	<i>Costruzione e collaudo</i>	in base alle norme IEC
<i>Cuscinetti</i>	schermati lubrificati a vita	<i>Tensione in uscita</i>	0-50 Volt

Scheda tecnica del motore e della dinamo tachimetrica

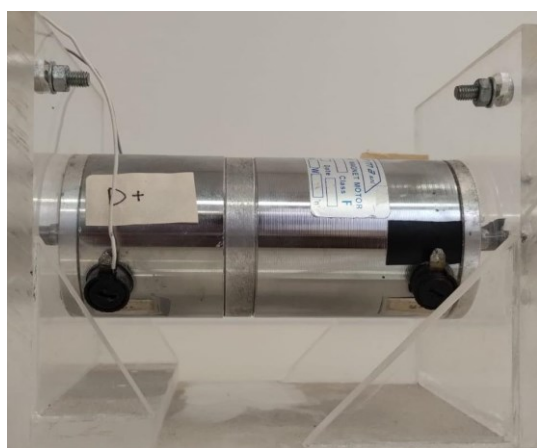


Figura 3: Vista frontale del motore.



Figura 4: Vista dall'alto del motore.

1.3.2 Amplificatore a mosfet

Viene usato un amplificatore a mosfet con modulatore PWM per amplificare il segnale di controllo proveniente dalla scheda MicroDAQ e darlo in ingresso al motore. L'amplificatore viene alimentato a corrente continua a 24V.

<i>Amplificatore a commutazione mosfet con modulatore PWM</i>	
<i>Tensione nominale</i>	24V \pm 10%
<i>Corrente nominale</i>	8 A
<i>Frequenza di commutazione</i>	50 KHz
<i>Ingresso analogico</i>	\pm 5V

Scheda tecnica dell'amplificatore a mosfet.

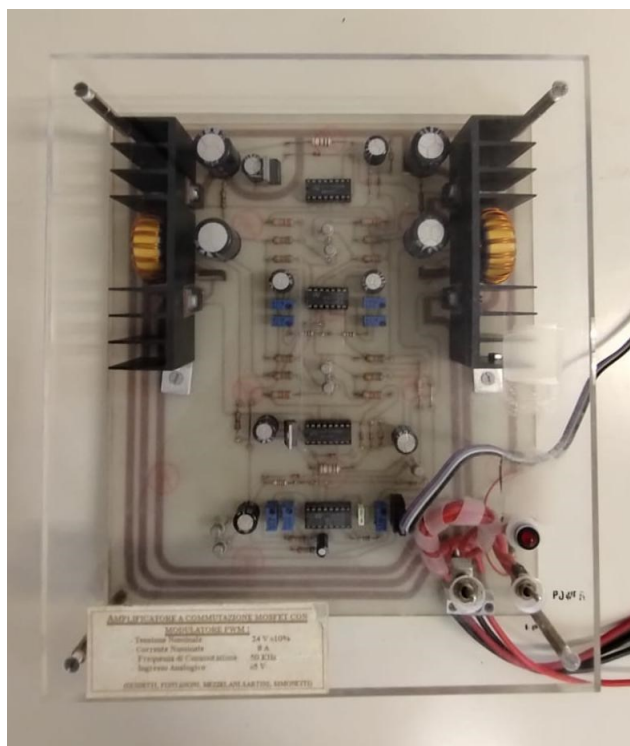


Figura 5: Amplificatore a mosfet.

1.3.3 Partitore di tensione con condensatore

Il partitore di tensione, necessario per proteggere la scheda, è composto da due resistenze, $R_1=120\text{ KOhm}$ e $R_2=15\text{ KOhm}$, con lo scopo di ridurre la tensione in uscita dalla dinamo che è variabile da 0-50 Volt e di conseguenza può essere superiore alla tensione massima che si può dare all'ingresso analogico della scheda. La tensione in uscita dalla dinamo viene divisa di un rapporto pari a 9. Il condensatore ha lo scopo di eseguire una prima filtrazione della tensione in uscita dalla dinamo in quanto questa non è un valore costante dipendente solo dalla velocità di rotazione per via delle caratteristiche intrinseche della dinamo e inoltre è disturbata da rumore.

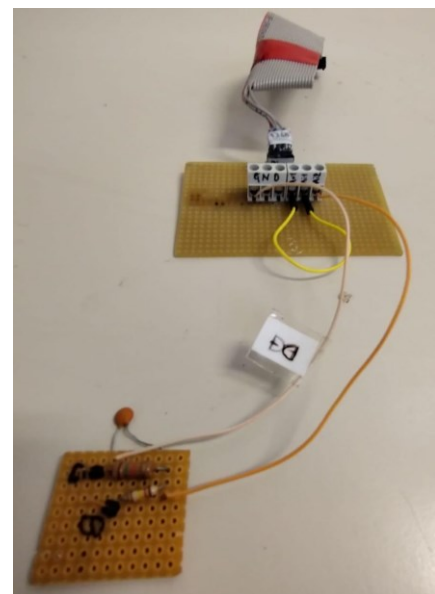
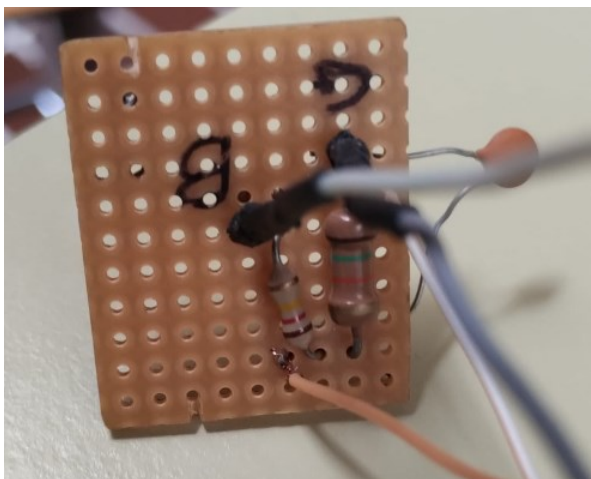


Figura 6 e figura 7: Partitore di tensione con condensatore a sinistra e partitore collegato al circuito che permettono il collegamento alla scheda MicroDAQ e alla dinamo.

1.3.4 Scheda MicroDAQ E2000 AC05-DAC05-12

MicroDAQ E2000 è una scheda DSP finalizzata all'acquisizione alla manipolazione di dati in tempo reale, può essere quindi usata per tutte le applicazioni che controllano e processano segnali digitali.

Questo dispositivo dispone di una serie di ingressi e uscite configurabili in modi diversi, ci sono ingressi e uscite analogiche, ingressi e uscite digitali, canali di PWM (pulse width modulation), moduli encoder e UART.

Nello specifico viene usata la scheda MicroDAQ E2000 AC05-DAC05-12 equipaggiata da 4 processori indipendenti e 32 Gb di memoria di archiviazione interna.



Figura 8: Vista pin scheda.



Figura 9: Vista frontale scheda.



Figura 10: Scheda MicroDAQ.

Scheda Tecnica MicroDAQ E2000 AC05-DAC05-12

<i>Processore</i>	456Hz TI C674x fixed/floating point DSP core 456Hz ARM926EJ-S RISC MPU 227 PRU 32bit RISC core 227 PRU 32bit RISC core
<i>Disco</i>	32 Gb
<i>Connettività</i>	Ethernet 100Base-TX WIFI, IEEE 802.11n, connettore RP-SMA, antenna 9dBi, USB2.0 450Mbit (dispositivo di archiviazione di massa)
<i>I/O digitale</i>	16 DIO 5V configurabili in: -6 PWM -1 UART -2 Quadrature encoder
<i>Input analogico ADC05</i>	Numero di canali: 16 Risoluzione: campionamento simultaneo a 16bit Frequenza di campionamento: 600ksps max Protezione contro la sovratensione Range di input programmabili: $\pm 5V$ o $\pm 10V$ Input single-ended Accuratezza relativa: $\pm 1.5LSB$ INL
<i>Output analogico DAC05</i>	Numero di canali: 16 Risoluzione: campionamento simultaneo a 16bit Range di output: 0-5V, 0-10V, $\pm 2.5V$, $\pm 5V$, $\pm 10V$ Output drive: 10mA Accuratezza relativa: $\pm 4LSB$ INL
<i>Alimentazione</i>	5V DC USB
<i>Temperatura</i>	0° C fino +70° C (se operativo), -40° C fino a +90° C (se usato come dispositivo di archiviazione)
<i>Dimensioni</i>	53.5x131x172 mm

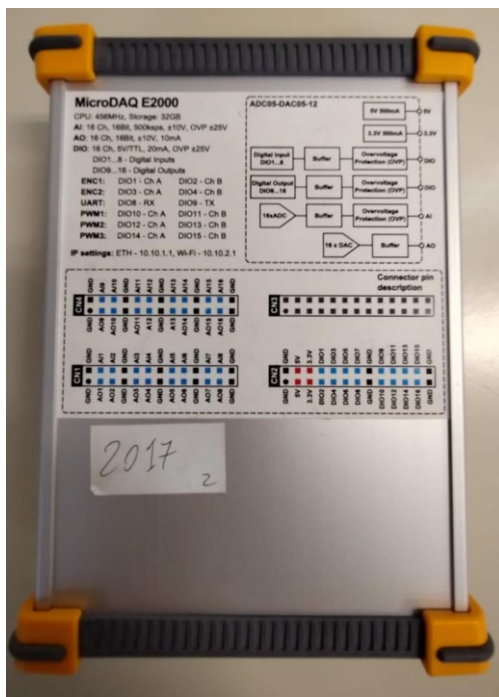


Figura 11: Vista scheda dall'alto.

Dalla figura 11 si può vedere la struttura interna del hardware della scheda e la disposizione dei pin.

Inoltre, tutti gli ingressi e le uscite digitali, e gli ingressi analogici, sono protetti da dispositivi di protezione per evitare che siano danneggiati da tensioni troppo alte.

Il software della scheda permette di usare il dispositivo con diversi programmi come Scilab, Matlab, LabView e con alcuni linguaggi di programmazione come C/C++ e Python.

È presente, inoltre, un web panel integrato che permette azioni di manutenzione della scheda insieme all'accesso alle impostazioni e alla memoria interna del dispositivo. È possibile anche effettuare acquisizioni di dati direttamente dal pannello web e impostare l'auto avvio della scheda per certe applicazioni DSP.

Il pannello è accessibile da qualsiasi browser inserendo l'indirizzo ip della scheda nella barra di ricerca come se fosse un indirizzo web.

La scheda può immagazzinare 32 Gb di acquisizioni e altri dati utente, accessibili con connessione USB o dal web panel.

1.4 Struttura completa sistema

La struttura completa del sistema risulta la seguente: il motore che si desidera controllare è alimentato dall'amplificatore a mosfet, a sua volta alimentato a 24 Volt. L'amplificatore mosfet fornisce più o meno alimentazione al motore in base al segnale di controllo, in uscita dalla scheda MicroDAQ, variabile tra -5 e 5 Volt. Il motore presenta in unico blocco anche la dinamo tachimetrica che produce una tensione variabile tra 0 e 50 Volt proporzionale alla velocità di rotazione del motore, la tensione in uscita dalla dinamo viene ridotta dal partitore di tensione e filtrata dal condensatore, solo allora può essere data in ingresso alla scheda insieme al segnale di controllo generato dalla scheda stessa. Complessivamente vengono usate due uscite analogiche e un ingresso analogico della MicroDAQ.

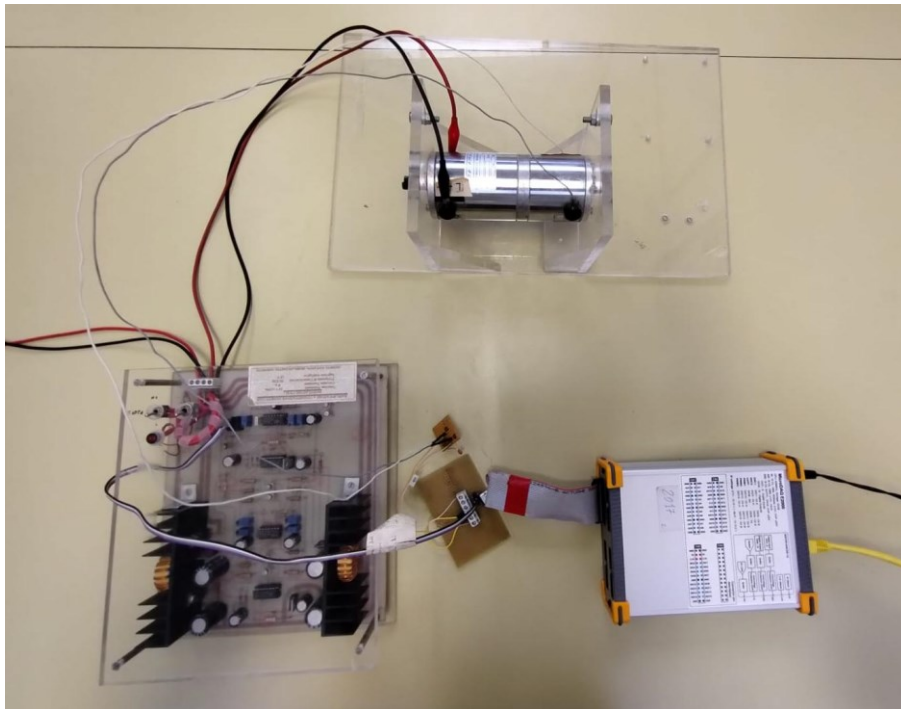


Figura 12: sistema completo

1.5 Software usato

I software usati sono: Scilab e la corrispettiva applicazione Xcos, Matlab e la corrispettiva applicazione Simulink.

Matlab è un ambiente di sviluppo e linguaggio di programmazione usato per il calcolo numerico e l'analisi di dati, in questa tesi viene usato per diverse applicazioni durante l'acquisizione dei dati per stimare il processo e durante la progettazione del controllore.

Simulink è un ambiente di modellazione e simulazione grafica specializzato nella modellazione, simulazione e analisi di sistemi dinamici, sia continui che discreti. Viene usato per simulare il controllo del sistema in catena chiusa.

Scilab è ambiente di programmazione e calcolo numerico open-source simile a Matlab. Viene usato per interfacciarsi con la scheda MicroDAQ tramite il toolbox Microdaq.

Xcos, parte di Scilab, è un ambiente di modellazione e simulazione grafica simile a Simulink di Matlab. Viene sfruttato per modellare il sistema di rivelazione e il sistema di controllo tramite un diagramma a blocchi, il quale verrà automaticamente convertito in codice C/C++, compilato, e caricato sulla scheda.

MicroDAQ, infatti, ha sviluppato un toolbox per Scilab che permette di sviluppare applicazioni avanzate per l'acquisizione e il controllo dati real-time direttamente su Scilab e XCos senza il bisogno di convertire manualmente dati o codice di alcuni tipo. I dati possono essere quindi visualizzati e manipolati direttamente, ciò si può fare scrivendo codice Scilab con le funzioni implementate dal toolbox o lavorare creando diagrammi su XCos dal quale verranno generate direttamente le applicazioni DSP e caricate automaticamente sulla scheda tramite wi-fi o ethernet.

Il toolbox rende a disposizioni diversi blocchi che possono essere usati per la modellazione tramite XCos e permettono l'accesso al hardware della scheda. C'è anche la possibilità di usare blocchi definiti dall'utente contenenti codice in C/C++ per una esecuzione ancora più veloce e ottimizzata. Al fine di poter

generare automaticamente il codice C/C++ destinato alla scheda è necessario il compilatore *C6000 DSP compiler* della Texas Instruments. Per Matlab servono altri due programmi sempre della Texas Instruments, *Real-Time Operating System* e *XDCTools*. È necessario che i tre programmi sopracitati siano a delle versioni specifiche.

1.6 Applicazioni standalone ed external-mode

Le applicazioni DSP sviluppate per la scheda possono essere di due tipi, standalone ed External Mode.


Le applicazioni standalone contengono solo comandi da eseguire in real-time e task di alta priorità e non c'è comunicazione con l'ambiente XCos, mentre le applicazioni sviluppate in External Mode permettono un accesso live ai dati analizzati e oltre ai task di alta priorità hanno task di bassa priorità proprio per la comunicazione con XCos. Il codice modellato tramite XCos, per esempio, sarà real-time e ad alta priorità mentre la comunicazione con XCos è appunto a bassa priorità e avviene solo quando si sono conclusi i task ad alta priorità.


La scheda DSP MicroDAQ verrà usata in external mode al fine di poter visionare i segnali con cui si sta lavorando direttamente dall'ambiente XCos.

1.7 Blocchi XCos

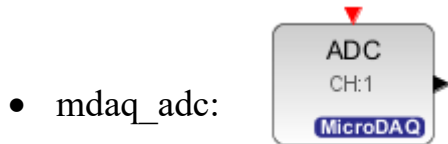
Di seguito sono mostrati e spiegati i blocchi usati per l'acquisizione dei dati relativi al sistema preso in considerazione e per il controllo del sistema stesso.

I blocchi nei quali nomi compare 'mdaq' sono blocchi introdotti dal toolbox di MicroDAQ, tali blocchi permettono l'accesso dell'utente alle periferiche hardware della scheda.

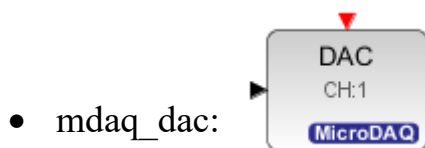
- mdaq_setup: 
configura le proprietà del modello sviluppato con XCos, è un blocco necessario e permette di specificare la durata dell'esecuzione del modello desiderato. Permette di scegliere la modalità di esecuzione tra standalone ed external mode, il tipo di compilazione tra debug e release con conseguenze sull'ottimizzazione dell'applicazione, l'opzione profiling che permette di vedere i tempi di esecuzione del modello e il tipo di risolutore da usare per i modelli a tempo continuo.

- CLOCK_c: 
configura il tempo di campionamento e il tempo di inizializzazione del modello, questo blocco è fondamentale perché genera eventi

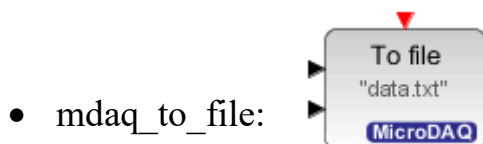
temporizzati regolari che vanno in input a tutti i blocchi che hanno bisogno di una temporizzazione precisa.



questo blocco legge e configura gli input analogici. Permette di scegliere i canali di input, il range dell'input, il tipo di misurazione e la possibilità di migliorare la accuratezza dell'input di ingresso tramite l'averaging, facendo la media da un numero di 4 a 256 campioni ADC. Prende come input gli eventi generati dal blocco clock_c e ha come output il segnale che legge in ingresso dalla periferica hardware.




questo blocco scrive e configura gli output analogici. Permette di scegliere i canali di output, il range dell'output, il valore iniziale e il valore finale. Prende come input gli eventi generati dal blocco clock_c per la sua temporizzazione e il segnale da mandare in output.

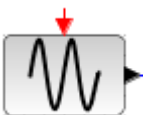


questo blocco ha il compito di salvare i dati desiderati su file in formato di file testo o file binario e permette di salvare i file in modo sequenziale


se la mole di dati è grande. Prende in input gli eventi generati dal blocco `clock_c`, i dati da salvare su file, e un trigger per segnalare al blocco quando salvare i file in modo sequenziale. I dati salvati sono nella memoria della scheda MicroDAQ nella cartella 'dsp/data/' e sono accessibili anche con la funzione `mdaqFileData`.

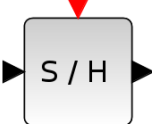
- `mdaq_signal`: 


questo blocco è necessario per leggere i segnali dell'applicazione DSP generati dal diagramma di XCos, per esempio prima di visualizzarli su uno scope, un oscilloscopio virtuale. Prende in ingresso il segnale dell'applicazione DSP e dà in uscita un segnale elaborabile e visualizzabile in Scilab. Ogni blocco `mdaq_signal` deve essere identificato da un id univoco.


- `mdaq_sinus`: 

questo blocco genera sinusoidi di ampiezza, frequenza, fase, ritardo e offset desiderato, prende in input gli eventi temporali generati dal blocco `clock_c` e dà output il segnale sinusoidale desiderato.


- mdaq_square:  questo blocco genera un'onda quadra di ampiezza, periodo, durata dell'impulso, ritardo e offset desiderato, prende in input gli eventi temporali generati dal blocco clock_c e dà in output il segnale desiderato.

- samphold_m:  questo blocco effettua una operazione di campionamento e tenuta, ogni volta che un evento viene ricevuto in input dal blocco questo viene copiato e tenuto in output fino all'arrivo dell'evento successivo. Questo blocco deve ricevere gli eventi temporali del blocco clock_c.


- cscope:  questo blocco è un oscilloscopio virtuale che permette di visualizzare graficamente i dati che riceve in input, necessita degli eventi temporali del blocco clock_c.

- saturation:  questo blocco impone un limite inferiore e superiore ai segnali che lo attraversano, viene usato per garantire che non arrivino tensioni superiori

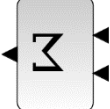
a quelle massime consentite alla scheda MicroDAQ per evitare che possa danneggiarsi.

- const_m: 

blocco costante che dà in output un valore costante nel tempo.

- mux: 

multiplexer che riceve più segnali e li instrada in unico segnale.

- summation: 

blocco sommatore.

- gainblk: 

blocco di guadagno.

- clr: 

questo blocco realizza un sistema lineare SISO rappresentato dalla funzione di trasferimento.

La struttura di un'applicazione dsp per la scheda microDAQ modellata con un diagramma di Xcos deve essere la seguente:

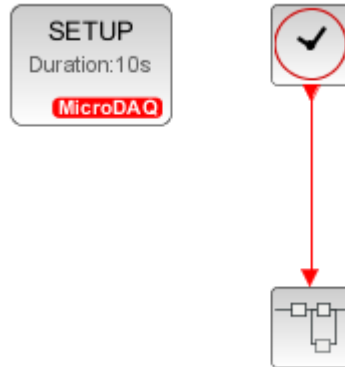



Figura 13: Struttura applicazione Xcos per MicroDAQ.

Il superblocco  è un blocco speciale che contiene un diagramma a blocchi e collegamenti agli ingressi e uscite del diagramma che rappresenta. Per entrare nel diagramma dentro al superblocco è sufficiente cliccare il blocco due volte.

Per sviluppare applicazioni per microDAQ il superblocco deve contenere tutti i blocchi del diagramma esclusi il blocco `mdaq_setup` e il blocco `clock_c` in quanto è proprio il diagramma dentro al superblocco che verrà usato per la generazione del codice C/C++ da caricare sulla scheda.

Capitolo 2

Identificazione

2.1 Identificazione del processo da controllare

Si desidera controllare il motore elettrico in velocità, per fare ciò si utilizza come modalità di controllo il controllo a retroazione. Il sistema da controllare in feedback, tuttavia, non è composta dal solo motore con dinamo annessa ma anche dal partitore di tensione, dal filtro costituito dal condensatore e dall'amplificatore a mosfet che amplifica il segnale di controllo in ingresso al motore. Considerando che il sistema è composto da più componenti di cui non si conoscono tutti i parametri costruttivi si procede all'identificazione del modello matematico. Viene usato un modello gray box, ovvero un modello che combina considerazioni teoriche sulla struttura del modello insieme a dati sperimentali da cui estrapolare una stima del modello completo.

Vengono eseguiti due tipi di identificazione:

- Risposta in frequenza o risposta armonica
- Risposta indiciale

Per entrambe le identificazioni viene usato lo stesso diagramma XCos 'acquisizione.zcos' andando a cambiare solamente il blocco che genera la funzione in ingresso al blocco DAC, ovvero il segnale in ingresso

all'amplificatore. Nel caso dell'identificazione in frequenza si usa il blocco mdaq_sinus, andando a generare in ingresso una senoide, mentre per la risposta indiciale il blocco mdaq_square che genera un ingresso a gradino.

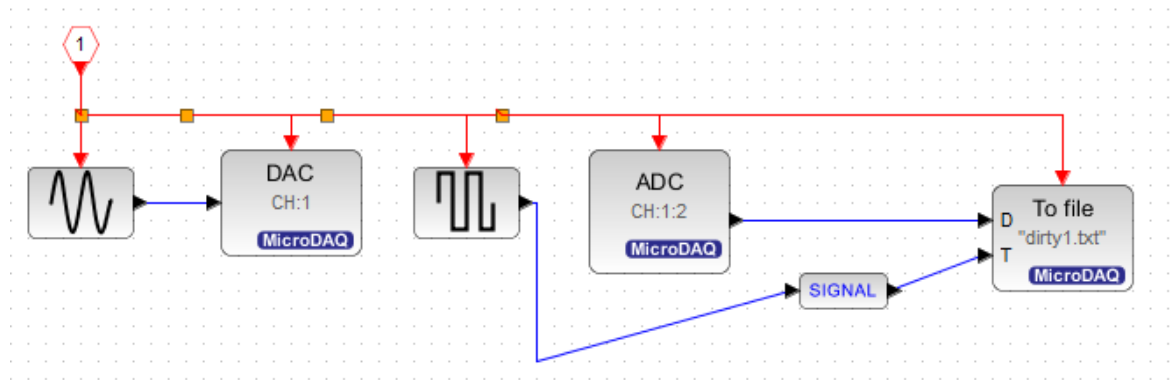


Figura 14: Applicazione acquisizione.zcos.

Il segnale viene generato dal blocco mdaq sinus o square ed entra in input al blocco mdaq_dac, viene quindi generato il segnale analogico dal pin AO1 e questo viene amplificato dall'amplificatore mosfet e va ad alimentare il motore elettrico. Lo stesso segnale non amplificato viene allo stesso tempo dato in input alla scheda sul pin AI1.

Il segnale in uscita dalla dinamo, opportunamente ridotto e filtrato arriva in input alla scheda sul pin AI2 ed elaborato tramite il blocco mdaq_adc. Il blocco mdaq_daq viene impostato per ricevere dati in ingresso su due canali, 1 e 2. Sul primo canale arriva il segnale generato dalla scheda stessa per poter essere confrontato con il segnale in uscita dal partitore che arriva sul secondo canale.

Il blocco mdaq_adc manda i due segnali in ingresso al blocco mdaq_tofile per essere salvati e visionati successivamente. Un blocco mdaq_square viene usato per mandare un trigger al blocco mdaq_tofile per iniziare il salvataggio, tale trigger per essere letto deve passare per il blocco mdaq_signal. I segnali così ricevuti vengono quindi salvati sulla scheda MicroDAQ nella posizione 'dsp/data/'.

Siccome al fine di avere un'applicazione DSP per la scheda MicroDAQ il diagramma XCos deve essere composto solamente dal blocco di setup, dal blocco clock_c, e dal superblocco, il diagramma dell'applicazione acquisizione.zcos nell'immagine sopra viene raccolto in un super blocco e l'applicazione XCos risultante ha la forma come nella *figura 13*.

Nel blocco clock_c viene impostato il periodo di campionamento a 1 kHz con tempo di inizializzazione nullo mentre nel blocco mdaq_setup viene impostata la durata dell'esecuzione dell'applicazione, la quale è variabile in base alla frequenza delle sinusoidi considerate, maggiore per frequenze basse e minore per frequenze alte. Durante l'identificazione con la risposta indiciale invece viene usata una durata di 3 secondi.

2.2 Risposta in frequenza

La risposta in frequenza, o risposta armonica, è una tecnica di identificazione che consiste nel sottoporre il sistema da stimare ad una serie di sollecitazioni sinusoidali di frequenza variabile e misurare la risposta a ogni sollecitazione. Si osserva quindi come il sistema risponde alle oscillazioni in ingresso in termini di ampiezza, fase e frequenza delle sue oscillazioni di uscita.

La risposta armonica è basata su una proprietà caratteristica dei sistemi lineari stazionari; se si applica ad un sistema LTI asintoticamente stabile un segnale di ingresso sinusoidale $x(t) = X\sin(\omega t)$, con una certa ampiezza X e pulsazione ω , a regime il sistema risponde con un segnale sinusoidale di medesima pulsazione ω che può essere espresso come $y(t) = Y(\omega)\sin(\omega t + \varphi(\omega))$.

L'ampiezza e la fase del segnale in uscita possono essere diversi da quella del segnale in ingresso per via delle caratteristiche del sistema e variano con la pulsazione ω del segnale in ingresso.

Viene definita la funzione di risposta armonica come:

$$F(\omega) = \frac{Y(\omega)}{X} e^{j\varphi(\omega)} = \frac{Y(\omega)}{X} (\cos(\varphi(\omega)) + j\sin(\varphi(\omega)))$$

$F(w)$ descrive completamente il comportamento del sistema a regime alle varie pulsazioni (o frequenze), inoltre la funzione di trasferimento $G(s)$ è legata alla funzione di risposta armonica $F(w) = G(s)|_{s=jw} = G(jw)$.

Si può trovare quindi per punti la funzione armonica $F(w)$, e quindi la funzione di trasferimento $G(s)$, trovando il valore del modulo e della fase per ogni pulsazione w considerata.

Per ogni frequenza quindi si valuta il rapporto tra le ampiezze massime dei due segnali, ovvero l'attenuazione di modulo, in dB per riportarla sul diagramma di Bode, e la fase della funzione di risposta armonica corrisponde allo sfasamento, in gradi, tra la sinusoide in ingresso e quella in uscita.

Ripetuto il procedimento per ogni frequenza considerata si possono tracciare i dati raccolti su scala logaritmica, ciò permette di ottenere i diagrammi di Bode del modulo e della fase della funzione di trasferimento del sistema.

Da tali diagrammi è possibile dedurre la formula matematica della funzione di trasferimento e quindi le caratteristiche cercate del sistema.

Tramite il programma *acquisizione.zcos* vengono generate funzioni sinusoidali di ampiezza 4V dal blocco mdaq_sinus campionate a 1000 Hz e delle seguenti frequenze in Hz:

0.05	0.01	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
0.9	1	1.5	2	2.5	3	3.5	4	4.5	5
5.5	6	6.5	7	7.5	8	8.5	9	9.5	10
10.5	11	11.5	12	12.5	13	13.5	14	14.5	15
15.5	16	16.5	17	17.5	18	18.5	19	19.5	20

Sotto i dati raccolti.

Frequenza [Hz]	Pulsazione [Rad/s]	Attenuazione [dB]	Sfasamento [gradi]
0,01	0,0628	0,8274	-0,3420
0,05	0,314	0,7856	-0,7920
0,1	0,628	0,7978	-1,2240
0,2	1,256	0,7791	-2,3760
0,3	1,884	0,7727	-3,5640
0,4	2,512	0,7721	-4,8960
0,5	3,14	0,7706	-5,7600
0,6	3,768	0,7722	-6,9120
0,7	4,396	0,8060	-7,5600
0,8	5,024	0,7861	-8,9280
0,9	5,652	0,7699	-9,3960
1	6,28	0,5600	-11,5200
1,5	9,42	0,6463	-14,5800
2	12,56	0,3329	-19,4400
2,5	15,7	0,2953	-22,5000
3	18,84	0,0072	-27,0000
3,5	21,98	0,1445	-30,2400
4	25,12	0,4637	-34,5600
4,5	28,26	0,6548	-37,2600
5	31,4	0,9191	-39,6000
5,5	34,54	-1,2326	-43,5600
6	37,68	-1,5353	-43,2000
6,5	40,82	-1,8621	-46,8000
7	43,96	-2,2101	-50,4000
7,5	47,1	-2,4092	-51,3000
8	50,24	-2,9321	-54,7200
8,5	53,38	-2,9940	-55,0800

9	56,52	-3,5919	-58,3200
9,5	59,66	-3,5512	-61,5600
10	62,8	-4,3379	-64,8000
10,5	65,94	-4,1414	-60,4800
11	69,08	-4,2438	-67,3200
11,5	72,22	-4,7380	-66,2400
12	75,36	-4,8408	-69,1200
12,5	78,5	-5,2049	-67,5000
13	81,64	-5,3686	-70,2000
13,5	84,78	-5,8039	-72,9000
14	87,92	5,8468	-75,6000
14,5	91,06	-6,1552	-73,0800
15	94,2	-6,4523	-70,2000
15,5	97,34	-6,7683	-78,1200
16	100,48	-6,9018	-74,8800
16,5	103,62	-7,3080	-77,2200
17	106,76	-7,1254	-79,5600
17,5	109,9	-7,6224	-81,9000
18	113,04	-7,8231	-77,7600
18,5	116,18	-7,9581	-79,9200
19	119,32	-8,2679	-82,0800
19,5	122,46	-8,4758	-84,2400
20	125,6	-8,7404	-79,2000

2.2.1 Dati raccolti

I dati raccolti sono disturbati da rumore e per essere analizzati al meglio vengono filtrati tramite un filtro Butterworth a fase zero codificato su Matlab nella funzione *filter_wave.m* che prende come ingresso la frequenza di campionamento, la frequenza dell'onda sinusoidale e un vettore contenente i due segnali da filtrare. La funzione ritorna un vettore contenente i due segnali filtrati, privi del rumore iniziale.


```

1 function [filtered_signal] = filter_wave(f,x)
2
3 fs = 1000; % Frequenza di campionamento in Hz, 1KHz di default
4
5 % Parametri del filtro
6 cutoff_freq = 4*f; % Frequenza di taglio per il filtro passa-basso (Hz)
7 filter_order = 3; %Ordine del filtro
8
9 normalized_cutoff_freq = cutoff_freq / (fs/2); %Frequenza di taglio normalizzata
10
11 [b, a] = butter(filter_order, normalized_cutoff_freq, 'low'); % Filtro Butterworth passa-basso
12
13 filtered_signal = filtfilt(b, a, x); % Filtraggio a fase zero per non sfasare le sinusoidi
14
15 end

```

Figura 15: Funzione filter_wave.m.

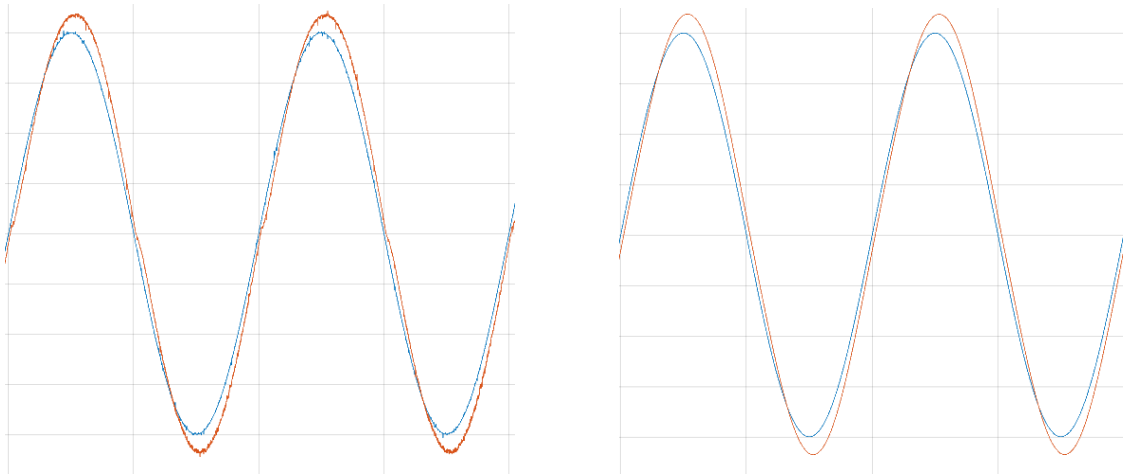


Figura 16 e figura 17: Onda sinusoidale disturbata da rumore a sinistra e a destra la stessa onda filtrata.

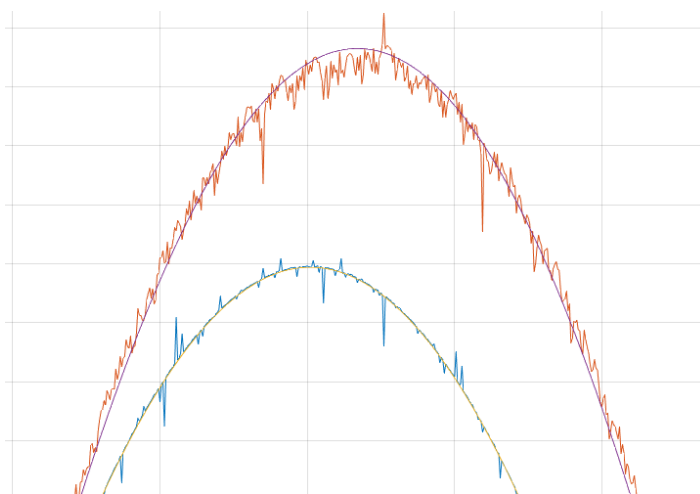


Figura 18: Dettaglio ingrandito, onda filtrata sovrapposta a un'onda disturbata.

I dati sperimentali filtrati vengono dati in ingresso alla funzione *analize_armonic_wave.m* tramite un vettore contenente le due sinusoidi. Specificata la frequenza dei segnali caricati e la frequenza con cui sono stati campionati i segnali la funzione permette di scegliere l'intervallo di campioni da plottare. Quando l'intervallo di campioni mostrato sul grafico è soddisfacente (sono isolati un solo massimo per ogni segnale considerato) la funzione trova l'attenuazione e lo sfasamento tra la sinusoidi di ingresso e quella di uscita.

```

1 function [mod_db,fase_grad] = analize_armonic_wave(vett)
2 %nella prima colonna di vett c'è il la funzione di ingresso,%nella seconda la funzione di uscita
3
4 Tc=0.001;
5 Tc=input('Periodo di Campionamento [sec]: //0.001 per 1 kHz: '); %di default è 1 KHz
6 freq_sin=input('Frequenza sinusoidi in ingresso[Hz]: ');
7
8 sizevett=size(vett); %dimensione del vettore che contiene le due
9
10 int1=1;
11 int2=sizevett(1);
12
13 figure
14 plot(vett(int1:int2,:)) %grafico intero della funzione registrata
15 drawnow
16 faiplot=input('altro grafico? s/n: ','s')
17
18 while faiplot=='s',
19     vett=vett(int1:int2,:);
20     int1=input('limite sx: '); %limite sinistro dei campioni considerati
21     int2=input('limite dx: '); %limite destri dei campioni considerati
22     plot(vett(int1:int2,:)) %grafico ristretto per evidenziare i massimi
23     faiplot=input('altro grafico? s/n: ','s') %per fare un nuovo grafico restringendo
24                                     %l' intervallo dei campioni considerati
25 end
26
27 [val,ind]=max(vett(int1:int2,:)) %trova i massimi della sinusoidi in ingresso ed in uscita
28 hold on
29 plot(vett(int1:int2,:));
30 plot(ind(1),val(1),'g*',ind(2),val(2),'g*') %evidenzia i massimi
31 hold off
32
33 rap_amp=val(2)/val(1);
34 mod_db=20*log10(rap_amp) %ampiezza in dB della risposta armonica, in corrispondenza
35 % della frequenza delle due sinusoidi
36 tmax1=(ind(1)*Tc);
37 tmax2=(ind(2)*Tc);
38 rit=(tmax2-tmax1)
39
40 fase_rad=rit*(2*pi)*freq_sin; %fase in radianti della risposta armonica
41 fase_grad=-(fase_rad*57.2958) %fase in gradi della risposta armonica, in corrispondenza della frequenza
42 %delle due sinusoidi con il segno meno perchè è un ritardo nel tempo (e quindi in
43 %fase) andrà compensato con una rete compensatrice
44 end

```

Figura 19: Funzione *analize_armonic_wave.m*.

Tali valori vengono quindi salvati in un vettore per potere essere rappresentati su un diagramma a scala semilogaritmica.

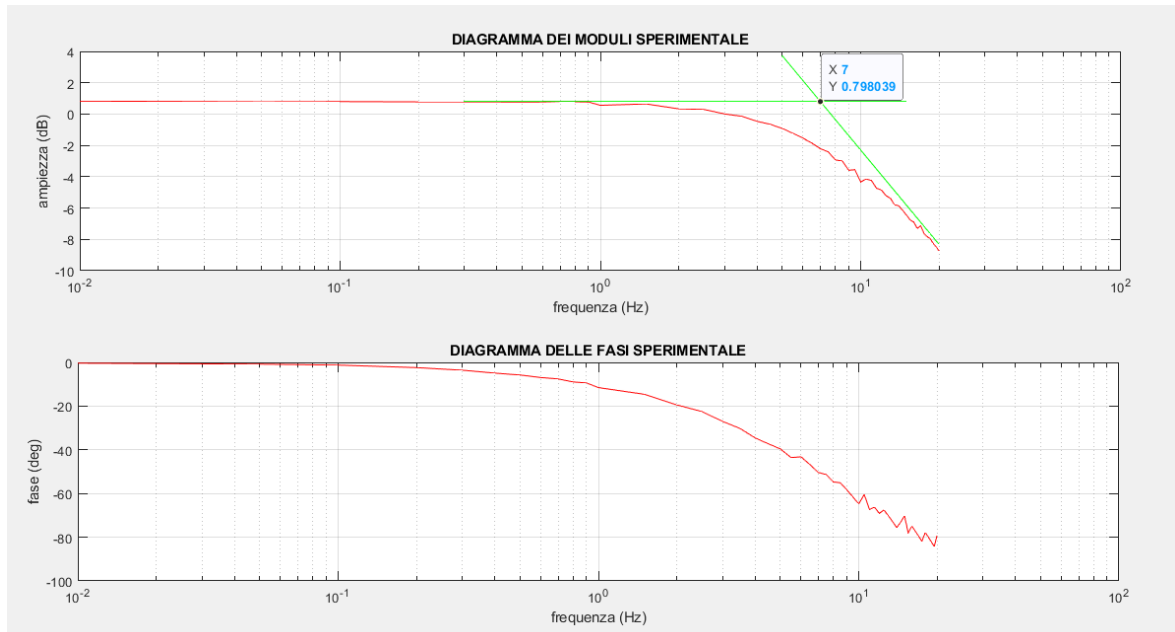


Figura 20: Plot dei dati raccolti su scala logaritmica.

Nel diagramma di Bode del modulo si possono vedere due tratti distinti, uno a 0 dB e uno a -20 dB. Andando a disegnare due rette tangenti a quei tratti si può trovare il polo cercato proprio nell'intersezione tra le due rette.

Si individua un polo intorno ai 44 rad/s ovvero circa 7 Hz.

Per calcolare la costante K di guadagno si osserva il valore del modulo alla frequenza più bassa, considerando che all'aumentare della frequenza il modulo e la fase diminuiscono sempre di più.

Si individua $K = 10^{\frac{0.8274}{20}} = 1.0999 \approx 1.1$.

0.8274 dB è il valore del modulo per $f = 0.01$ Hz ovvero $\omega \approx 0.063$ rad/s.

Il sistema identificato risulta quindi $W(s) = \frac{1.1}{1 + \frac{s}{44}}$

Il polo individuato è il polo meccanico, il polo elettrico si trova a pulsazioni più alte, di solito una decade più distante, e non è stato trovato. Viene fatta questa scelta perché il polo dominante è quello meccanico e per frequenze molto inferiori a quella del polo elettrico il sistema può essere approssimato con solo il polo meccanico.

A dimostrazione di ciò, la banda passante infatti risulta nell'intorno del polo meccanico e sperimentalmente si può vedere come, più si aumenta la frequenza della sinusoide di ingresso oltre la frequenza del polo meccanico individuato, più il sistema non risponde prontamente al segnale di ingresso fino a eventualmente ottenere la cessazione di rotazione del motore che, non potendo invertire la direzione di rotazione abbastanza in fretta inizia, a vibrare.

Viene quindi simulato il sistema $W(s) = \frac{1.1}{1 + \frac{s}{44}}$ e disegnato il suo diagramma di

bode per confrontarlo con il diagramma sperimentale.

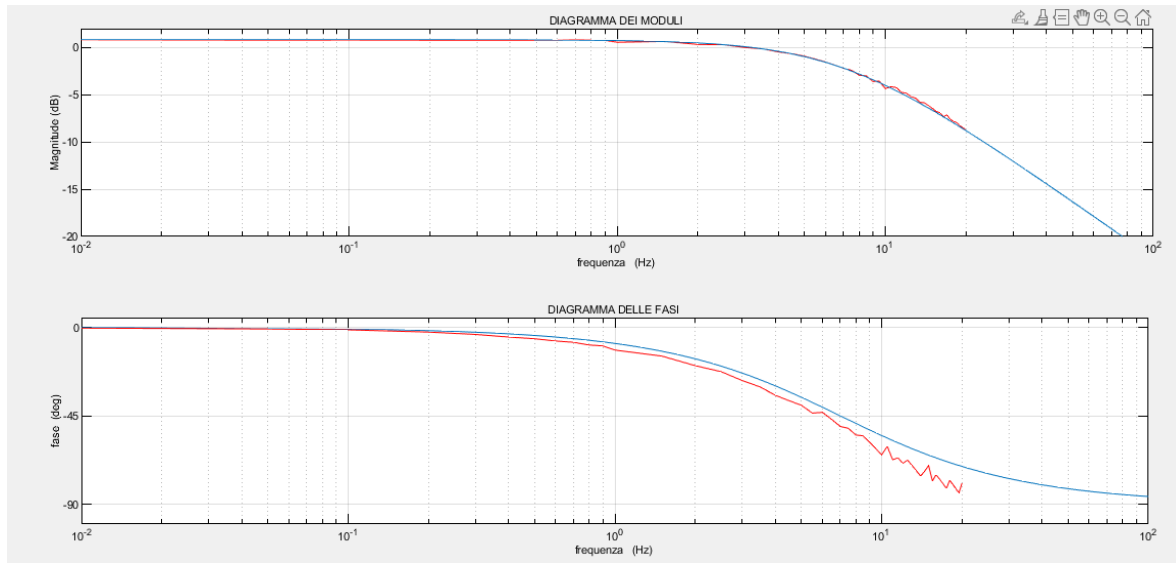


Figura 21: Diagrammi di Bode del processo stimato insieme ai dati raccolti.

Il grafico dei moduli mostra una buona corrispondenza tra la risposta armonica e la funzione di trasferimento stimata nell'intervallo delle frequenze analizzate. Nel diagramma delle fasi invece c'è più discrepanza tra i dati sperimentali e quelli stimati nelle frequenze più alte analizzate. La fase è minore nel diagramma sperimentale e ciò potrebbe indicare la presenza di un altro polo che spiegherebbe l'abbassamento della fase. Tuttavia, per le scelte di modellazione e rivelazione effettuate il polo in questione non viene cercato e individuato.

2.3 Risposta Indiciale

La risposta indiciale è un altro metodo di identificazione che consiste nel sottoporre il sistema ad un ingresso a gradino, misurarne la risposta e confrontarla con la risposta al gradino di un modello standard.

Il modello standard che si considera è quello che rappresenta un sistema del primo ordine nella forma $W(s) = \frac{K}{1+\tau s}$ con $k>0$ e $\tau>0$.

È stata effettuata un'identificazione con un gradino di ampiezza pari a 4 V.

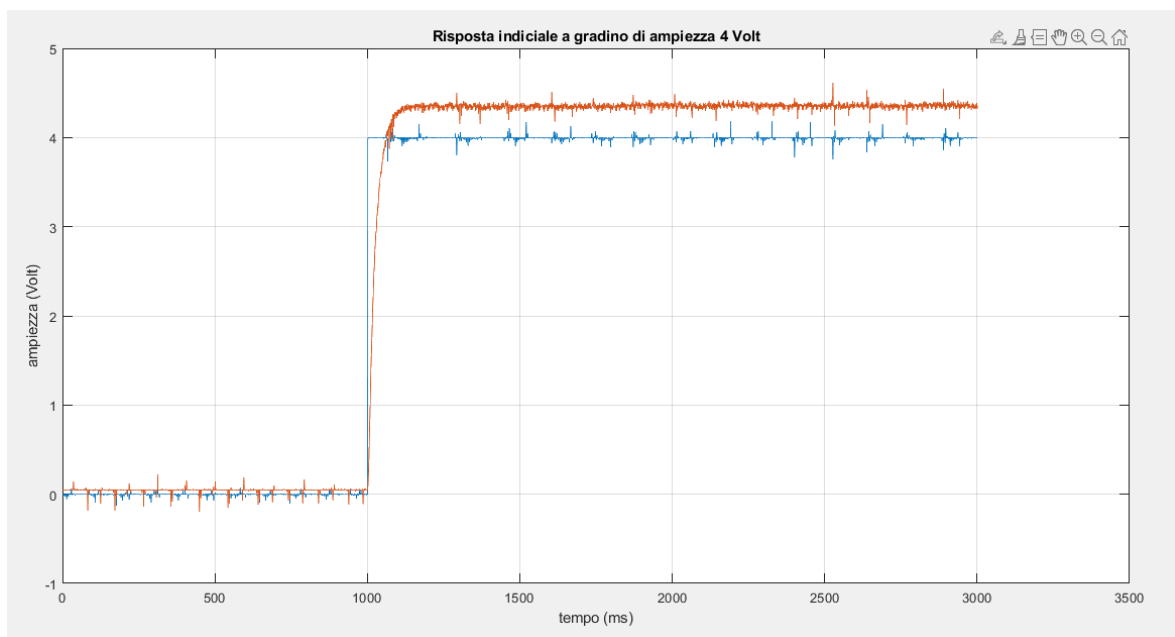


Figura 22: Risposta del sistema a un gradino di ampiezza 4V.

Dal grafico si vede che il valore di regime della risposta è maggiore del valore del gradino, ciò significa una costante $K > 1$. Essendo il valore di regime 4.36 V e l'ampiezza del gradino pari a 4 V otteniamo K pari a 1.09. $K = \frac{4.36}{4} = 1.09$.

Per i sistemi del primo ordine per $t = \tau$ la risposta assume un valore pari al 63,2% del valore finale di regime.

Facendo il procedimento inverso, $4.36 \cdot 0.632 = 2.756$, si verifica quindi che tempo t corrisponde al valore 2.756 V e otteniamo $t = 0.024$ s.

Possiamo stimare quindi il modello del sistema come $W(s) = \frac{1.09}{1+0.024s}$

A $\tau = 0.024$ corrisponde un polo di circa 41.67 rad/s ovvero circa 6.63 Hz.

Viene quindi simulato il sistema $W(s) = \frac{1.09}{1+0.024s} = \frac{1.09}{1+\frac{s}{41.67}}$

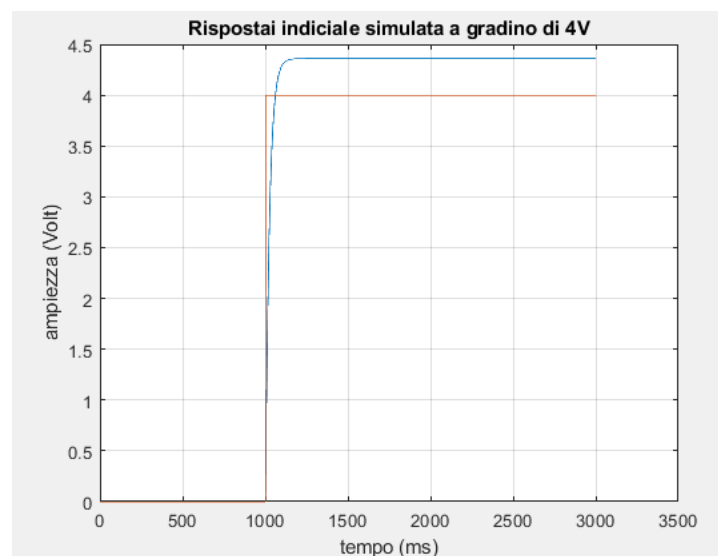


Figura 23: Risposta del sistema stimato a un gradino di ampiezza 4V.

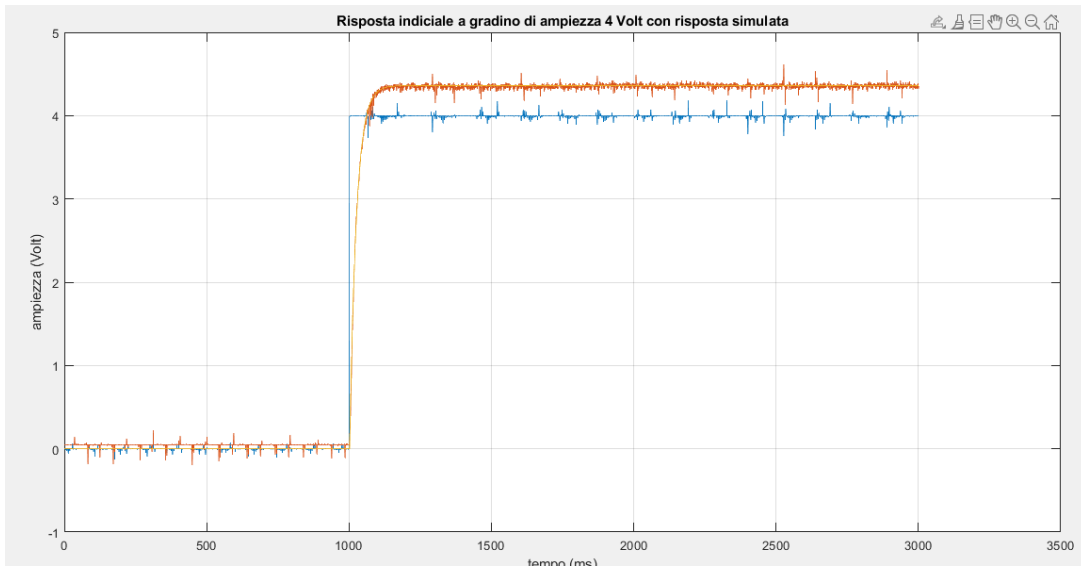


Figura 24: Risposta del sistema e del sistema stimato a un gradino di ampiezza 4V.

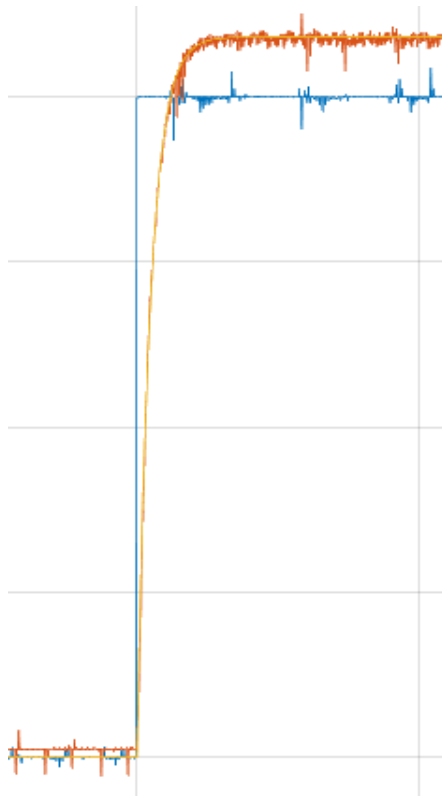


Figura 25: Dettaglio ingrandito della figura sopra.

I due metodi di identificazione forniscono due modelli matematici diversi.

$$\text{Risposta Armonica: } W(s) = \frac{1.1}{1+0.0227s} = \frac{1.1}{1+\frac{s}{44}}$$

$$\text{Risposta Indiciale: } W(s) = \frac{1.09}{1+0.024s} = \frac{1.09}{1+\frac{s}{41.67}}$$

Per la progettazione del controllore verrà considerato un modello intermedio tra i due:

$$W(s) = \frac{1.095}{1+0.0238s} = \frac{1.095}{1+\frac{s}{42}}$$

Capitolo 3

Progettazione controllore

3.1 Metodi e specifiche

Sono stati usati due metodi diversi per la progettazione del controllore per il controllo in velocità del motore: la sintesi in frequenza e la sintesi con il luogo delle radici.

Le specifiche considerate per la sintesi del controllore sono le seguenti:

- Errore a regime nullo a regime per ingressi a gradino
- Errore a regime inferiore al 4% per i disturbi a rampa
- Tempo di salita $t_s \leq 70\text{ms}$
- Sovraelongazione $< 5\%$

3.2 Sintesi in frequenza

La specifica sull'errore nullo a regime per ingressi a gradino si traduce in un controllore del tipo $G(s) = \frac{k}{s}$ visto che il processo stimato $P(s) = \frac{1.095}{1 + \frac{s}{42}}$ non ha poli nell'origine.

L'errore inferiore al 4% per ingressi a rampa si traduce in:

$$|e_1| = \frac{kd^2}{kf} = \frac{1}{kf} = \frac{1}{k*1.095} \leq 0.04 \text{ da cui } k \geq 22.93 \text{ e per sicurezza prendo}$$

$k=26$.

Il controllore di primo tentativo è $G(s) = \frac{26}{s}$

La specifica sul tempo di salita si traduce in una specifica sulla banda passante e quindi sulla pulsazione di attraversamento.

$$B_3 * t_s \approx 3 \longrightarrow \frac{3}{0.07} \approx 43 \text{ rad/s} \longrightarrow B_3 \geq 43 \text{ rad/s}$$

$$\text{inoltre } \omega_{t[\text{rad/s}]} \approx [0.5 \div 0.8] B_{3[\text{rad/s}]} \longrightarrow \omega_{t[\text{rad/s}]} \in [25, 40]$$

Prendo $\omega_t = 30 \text{ rad/s}$

La specifica sulla sovraelongazione $S \leq 2\%$ si traduce in una specifica sullo smorzamento e quindi sul margine di fase.

$$S\% = 100e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}} \text{ da cui } \xi = \frac{\ln\left(\frac{S\%}{100}\right)}{\sqrt{\ln\left(\frac{S\%}{100}\right)^2 + \pi^2}}$$

Calcolo alternativo:

$$S\% = 100e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}} = 100e^{-\tan(\beta)\pi} \leq S^*$$

$$\beta \geq \beta^* = \tan^{-1} \left\{ -\frac{1}{\pi} \ln \left(\frac{S^*}{100} \right) \right\} \text{ e } \xi = \sin(\beta)$$

Ricavo che $S\% \leq 2$ implica $\xi \geq 0.78$.

Considerando $m\varphi_{[\text{gradi}]} = 100 \cdot \xi$ ottengo $m\varphi \geq 59^\circ$

$$P(s) = \frac{1.095}{1 + \frac{s}{42}} \text{ e } G(s) = \frac{26}{s} \text{ da cui } FCap(s) = P(s)G(s) = \frac{26 \cdot 1.095}{s \cdot 1 + \frac{s}{42}}$$

I diagrammi di bode del sistema con controllore di primo tentativo sono:

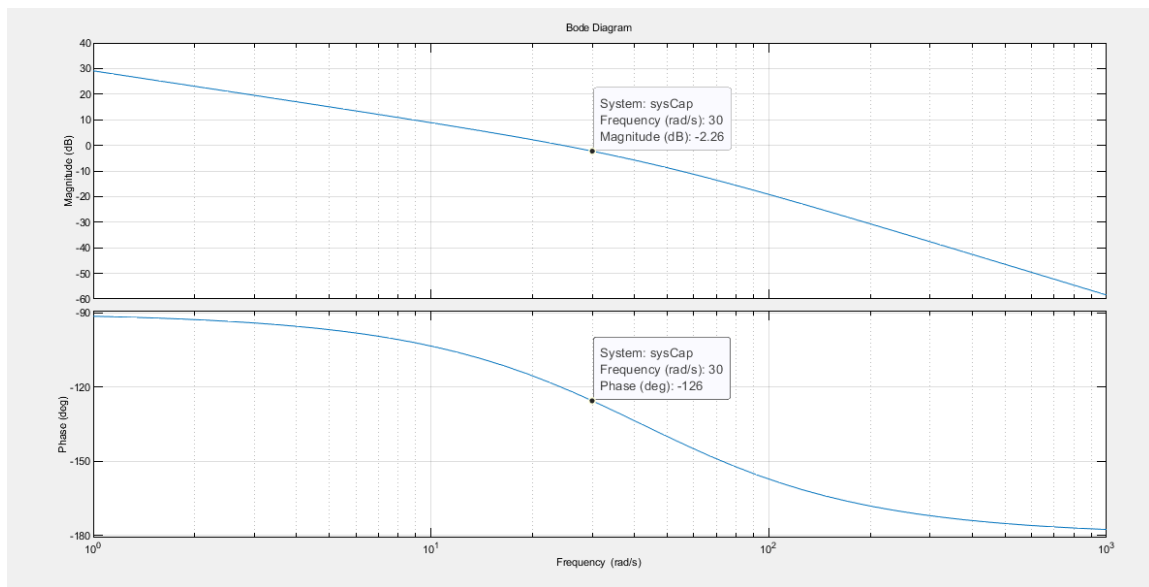


Figura 26: Diagrammi di Bode del processo con controllore di primo tentativo.

Si può vedere come in corrispondenza della pulsazione di attraversamento $\omega t = 30$ rad/s il modulo vale -2.26 db e la fase vale -126° con un conseguente $m\varphi = 180^\circ - 126^\circ = 54^\circ$

È necessario quindi alzare il modulo di 2.26 db e la fase di almeno 5 gradi in corrispondenza di ω_t .

Costruisco perciò una funziona anticipatrice, scelgo

$30\tau = 0.85$ da cui $\omega_a = 35.3$, prendo $m_a = 4$ e quindi $\omega_a * m_a = 141.2$

La funzione anticipatrice risulta $Ra(s) = \frac{1 + \frac{s}{35.3}}{1 + \frac{s}{141.2}}$.

Il controllore di secondo tentativo diventa $G(s) * Ra(s)$ da cui

$$F_{cap2} = F(s) * G(s) * Ra(s)$$

I diagrammi di bode risultanti sono:

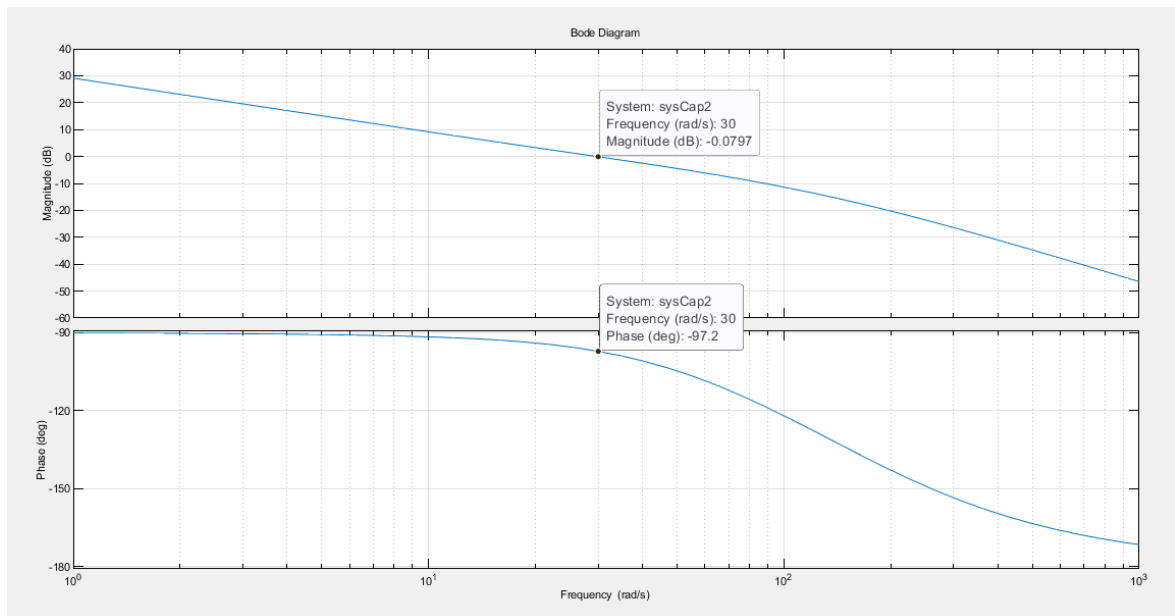


Figura 27: Diagrammi di Bode del processo con controllore di secondo tentativo.

Si può vedere come in corrispondenza della pulsazione di attraversamento $\omega=30$ rad/s il modulo vale -0.08 db, circa 0 db, e la fase vale -97.2° con un conseguente $m\varphi = 180^\circ - 97.2^\circ = 82.8^\circ$

Il controllore risulta $G(s) = \frac{26}{s} \frac{1 + \frac{s}{35.3}}{1 + \frac{s}{141.2}}$

Possiamo ritenere la sintesi conclusa.

3.2.1 Simulazione

Viene ora simulato il sistema controllato a feedback con il controllore sopra provato. La simulazione avviene su simulink e si dà in ingresso al sistema un segnale costante di ampiezza 4.

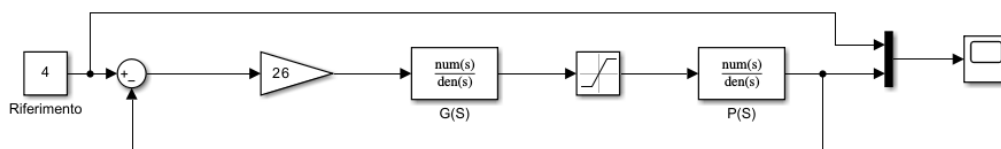


Figura 28: Diagramma in Simulink per simulare il controllo a retroazione.

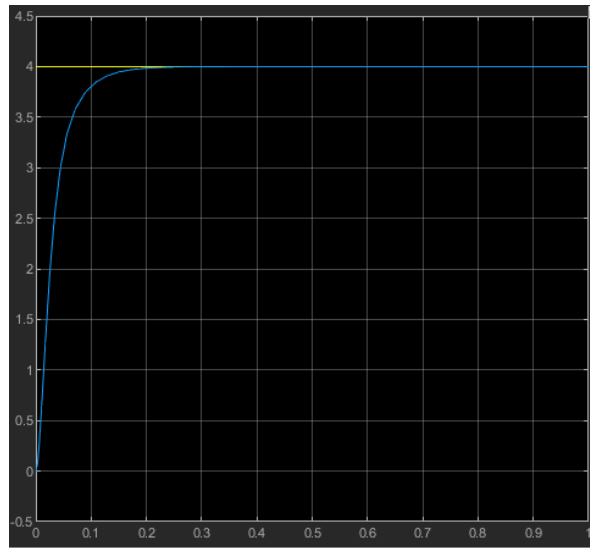


Figura 29: Risposta del sistema simulato.

La simulazione risulta soddisfacente con un tempo di salita di 65ms, un errore nullo a regime e sovraelongazione nulla.

Viene simulato anche un ingresso a rampa ottenendo un errore a regime di 0.035 che risulta soddisfacente alla specifica imposta.

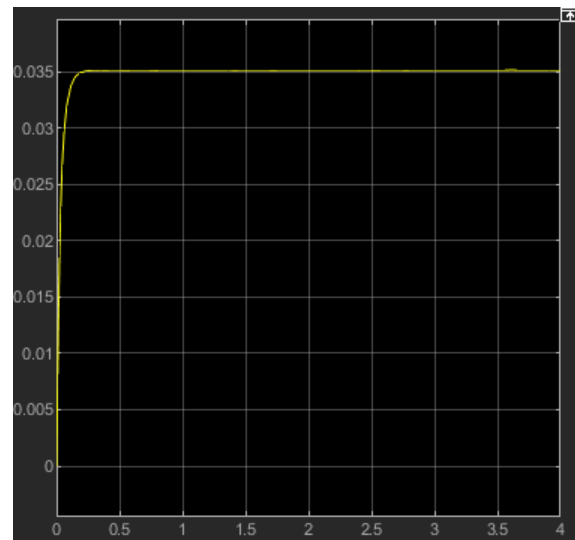


Figura 30 e figura 31: Risposta a un ingresso a rampa unitaria del sistema simulato a sinistra e errore tra ingresso e uscita a destra.

3.3 Sintesi con il luogo delle radici

La specifica sull'errore nullo a regime per ingressi a gradino si traduce in un controllore del tipo $G(s) = \frac{k}{s}$ visto che il processo stimato $P(s) = \frac{1.095}{1 + \frac{s}{42}}$ non ha poli nell'origine.

L'errore inferiore al 4% per ingressi a rampa si traduce in:

$$|e1| = \frac{kd^2}{kf} = \frac{1}{kf} = \frac{1}{k*1.095} \leq 0.04 \text{ da cui } k \geq 22.93$$

Il controllore di primo tentativo è $G(s) = \frac{1}{s}$

La specifica sulla sovraelongazione $S \leq 2\%$ si traduce in una specifica sullo smorzamento.

$$\text{Considero } S\% = 100e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}} \text{ da cui } \xi = \frac{\ln\left(\frac{S\%}{100}\right)}{\sqrt{\ln\left(\frac{S\%}{100}\right)^2 + \pi^2}}$$

Calcolo alternativo:

$$S\% = 100e^{-\frac{\xi\pi}{\sqrt{1-\xi^2}}} = 100e^{-\tan(\beta)\pi} \leq S^*$$

$$\beta \geq \beta^* = \tan^{-1}\left\{-\frac{1}{\pi}\ln\left(\frac{S^*}{100}\right)\right\} \text{ e } \xi = \sin(\beta)$$

Ricavo che $S\% \leq 2$ implica $\xi \geq 0.78$.

La specifica sul transitorio riferita al tempo di salita del tipo $t_s \leq T_s^*$ si traduce in una disuguaglianza su τ , diversa in base al grado del sistema.

$\tau \leq \frac{T_s^*}{1.8}$ per i sistemi del primo ordine con poli complessi coniugati

Da cui $r \geq \frac{1.8}{T_{sMax}}$ con $r = \frac{1}{\tau}$ che rappresenta la distanza dall'origine dei poli.

$T_s^* = 0.07$ e perciò $r \geq 25.7 \text{ rad/s} \approx 26 \text{ rad/s}$

Controllo ora il luogo delle radici per $FCap(s) = P(s)G(s) = \frac{1}{s} \frac{1.095}{1 + \frac{s}{42}}$

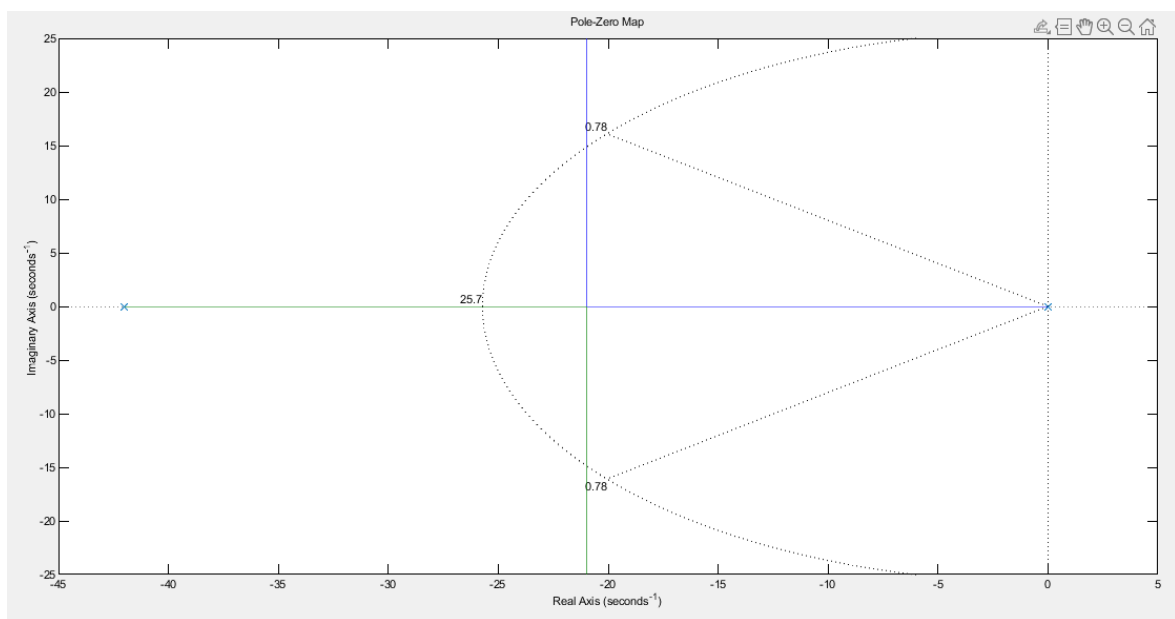


Figura 32: Luogo delle radici del processo con controllore di primo tentativo.

Le specifiche non sono soddisfatte ed è necessario spostare il centro degli asintoti nella zona di specifica. Uso una funzione compensatrice, uno zero e un polo.

Considerando che il centro degli asintoti è posizionato secondo la formula

$$\text{centro asintoti} = \frac{\sum \text{poli} - \sum \text{zeri}}{\text{numero poli} - \text{numero zeri}}$$

Avendo un polo a 42 rad/s e volendo spostare il centro degli asintoti nella zona di specifica oltre il punto a 26 rad/s usando una funzione compensatrice

$$R(s) = \frac{1 + \frac{s}{\text{zero}}}{1 + \frac{s}{\text{polo}}} \text{ ottengo } \frac{\sum \text{poli} - \sum \text{zeri}}{\text{numero poli} - \text{numero zeri}} = \frac{\text{polo} + 42 - \text{zero}}{2} \geq 26$$

Se posiziono lo zero a 30 rad/s trovo che il polo deve essere posizionato ad almeno 64 rad/s dal polo reale. Scelgo il polo a 65 rad/s e la funzione

$$\text{compensatrice risulta } R(s) = \frac{1 + \frac{s}{30}}{1 + \frac{s}{65}} .$$

$$\text{Controllo sul luogo delle radici } FCap2(s) = FCap(s) * R(s) = \frac{1}{s} \frac{1 + \frac{s}{30}}{1 + \frac{s}{65}} \frac{1.095}{1 + \frac{s}{42}} .$$

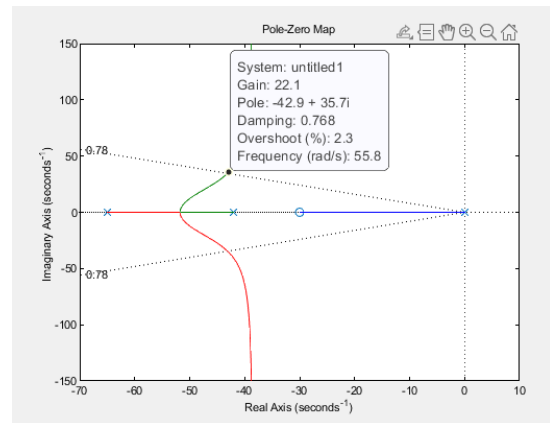
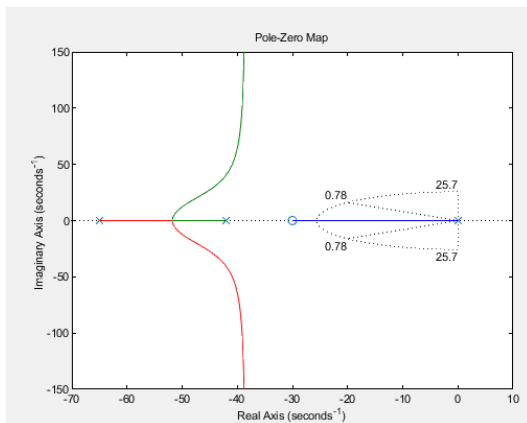


Figura 33 e figura 34: Luogo delle radici del processo con controllore di secondo tentativo.

Il centro degli asintoti si è spostato fuori dalla zona di specifica ma non è possibile scegliere un k soddisfacente che rispetti anche le specifiche sullo smorzamento.

Decido allora di spostare il polo più a sinistra, provo a prendere $p=90$ rad/s e verifico sul luogo delle radici. $R(s)$ diventa

$$R(s) = \frac{1 + \frac{s}{30}}{1 + \frac{s}{90}} \quad \text{e} \quad F_{Cap3}(s) = F_{Cap}(s) * R(s)$$

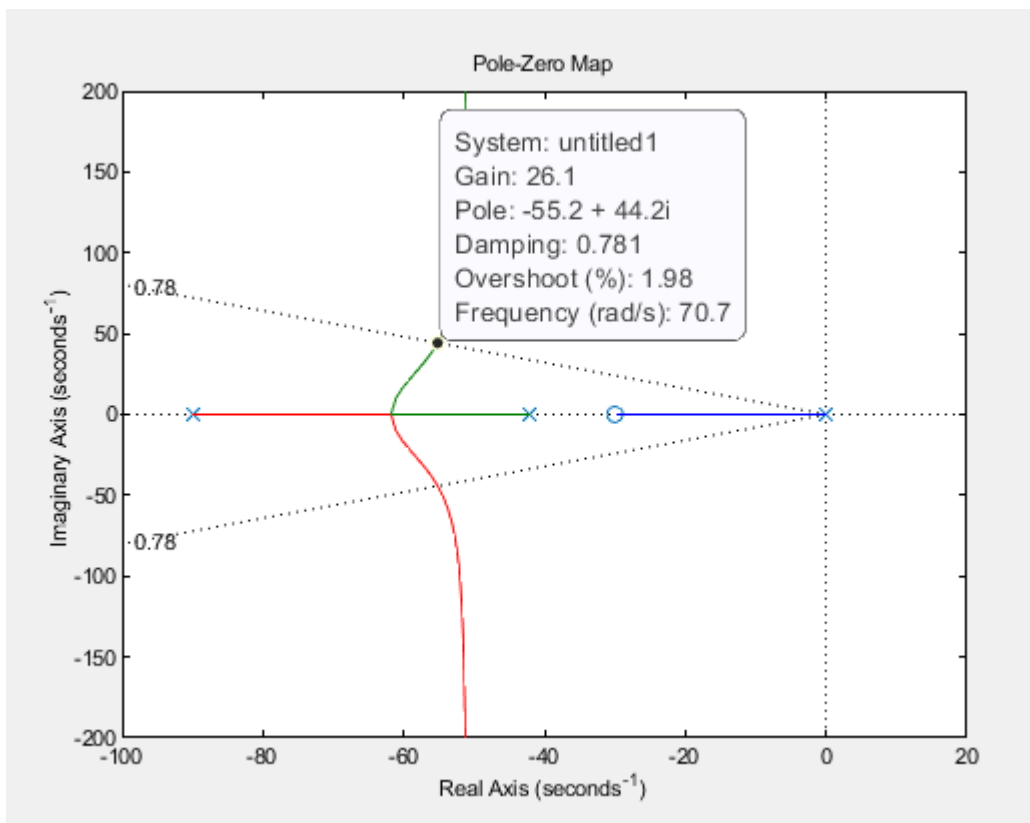


Figura 35: Luogo delle radici del processo con controllore di terzo tentativo.

Il centro degli asintoti è nella zona di specifica, k doveva essere maggiore di 22.83, riesco a prendere un k che soddisfa tutte le specifiche e scelgo $k=26$ per

avere un margine di errore e verifico che così rispetto le condizioni sullo smorzamento.

Il controllore risulta quindi $G(s) = \frac{26 \frac{s}{30} + 1}{s \frac{s}{90} + 1}$ e la sintesi può considerarsi conclusa.

3.3.1 Simulazione

Viene ora simulato il sistema controllato a feedback con il controllore sopra provato. La simulazione avviene su Simulink e si dà in ingresso al sistema un segnale costante di ampiezza 4.

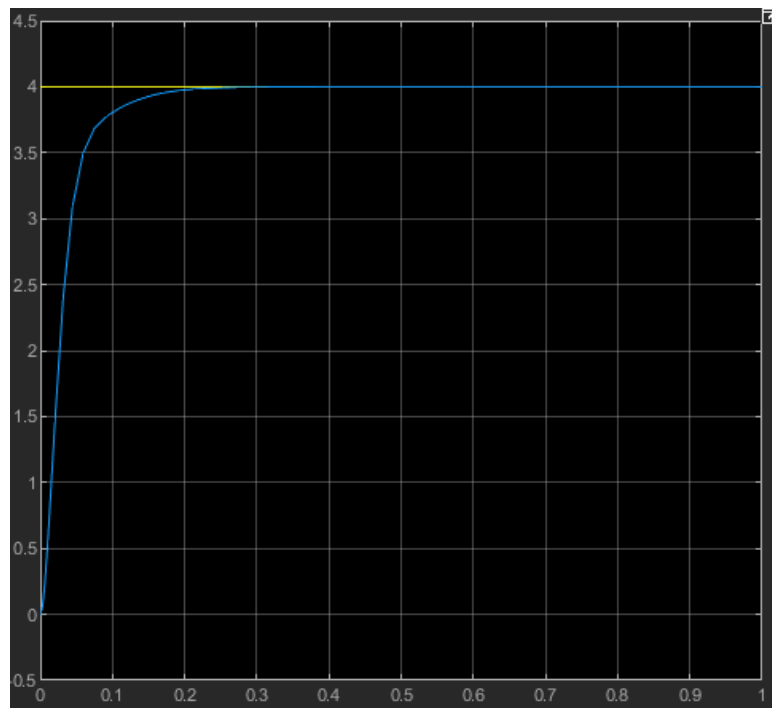


Figura 36: Risposta del sistema simulato ad un ingresso a gradino di ampiezza 4.

La simulazione risulta soddisfacente con un tempo di salita di salita di circa 60ms, un errore nullo a regime e sovraelongazione nulla.

Viene simulato anche un ingresso a rampa con un errore a regime di 0.035 che risulta soddisfacente alla specifica imposta.

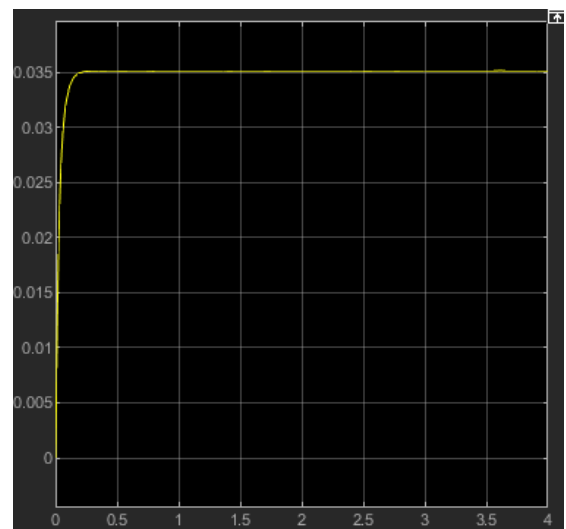


Figura 37 e figura 38: Risposta a un ingresso a rampa unitaria del sistema simulato a sinistra e errore tra ingresso e uscita a destra.

I due controllori sintetizzati sono quindi:

Sintesi in frequenza

$$G(s) = \frac{26 \frac{s}{35,3} + 1}{s \frac{s}{141,2} + 1}$$

Sintesi col luogo delle radici

$$G(s) = \frac{26 \frac{s}{30} + 1}{s \frac{s}{90} + 1}$$

Capitolo 4

Controllo

4.1 Controllo del motore elettrico

I due controllori progettati vengono quindi testati sul controllo del sistema vero e proprio.

Il controllo avviene tramite l'applicazione XCos progettata chiamata *controllo.zcos*

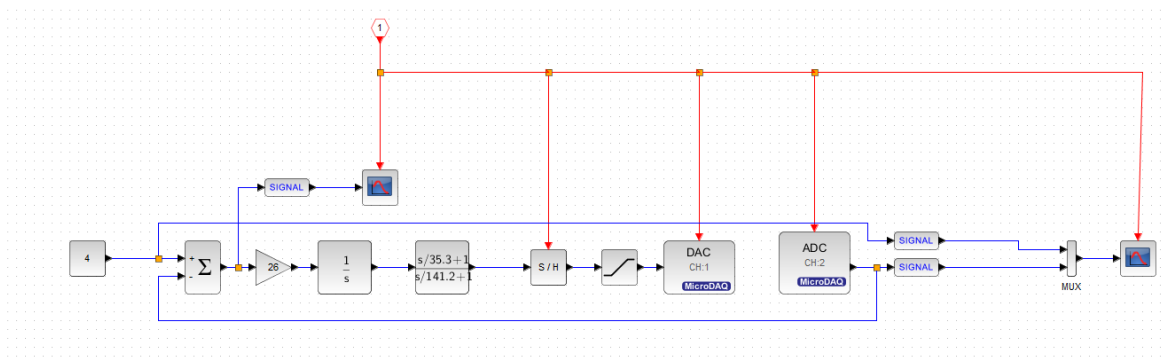


Figura 39: Diagramma XCos dell'applicazione *controllo.zcos*

L'applicazione *controllo.zcos* sviluppata modella un sistema a retroazione dove però a differenza di prima il processo non è più stimato e rappresentato dalla funzione di trasferimento ma è il motore elettrico insieme alle altre componenti hardware. L'ingresso al processo stimato viene quindi sostituito con l'ingresso

al blocco DAC che manda in output sul canale uno e quindi sul pin AO1 l'ingresso ricevuto. L'uscita invece è sostituita dal blocco ADC che riceve sul canale due e quindi sul pin AI2 il segnale in uscita dal partitore di tensione.

Vengono plottati sullo scope l'ingresso desiderato e la risposta del sistema.

Il diagramma nell'immagine *numero* viene raccolto in un superblocco e nel blocco clock_c viene impostato il periodo di campionamento a 1 kHz con tempo di inizializzazione nullo mentre nel blocco mdaq_setup viene impostata la durata desiderata dell'esecuzione.

4.2 Controllore progettato in frequenza

Il primo controllore che viene testato è quello costruito dalla sintesi in

frequenza, risulta essere $G(s) = \frac{26}{s} \frac{1 + \frac{s}{35.3}}{1 + \frac{s}{141.2}}$.

Viene dato in ingresso al sistema un gradino da 4 Volt e graficata sullo scope la risposta del sistema.

Si può notare un transitorio che non presenta sovraelongazione, il tempo di salita è inferiore a quello della risposta simulata con Simulink, circa 0.04 s.

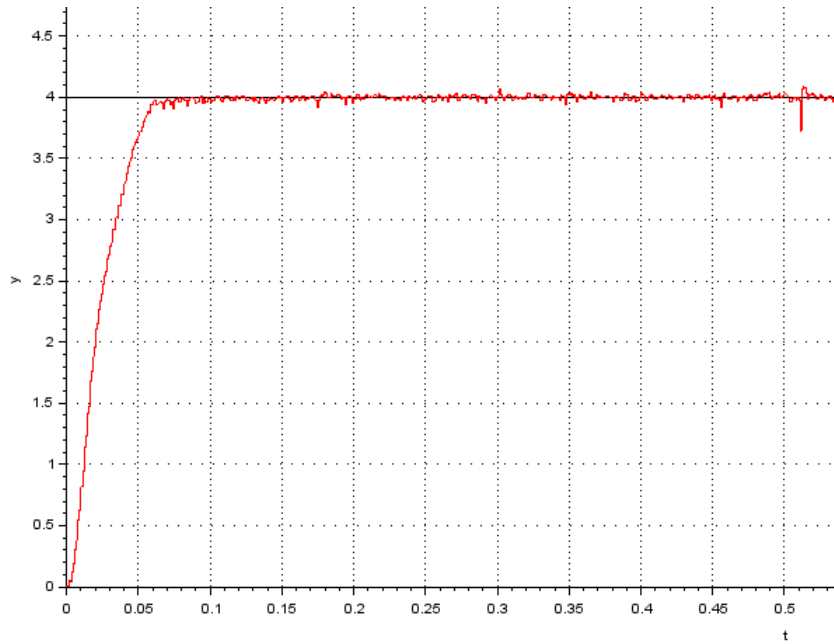


Figura 40: Risposta del sistema retroazionato a un ingresso a gradino da 4V.

Viene data in ingresso al sistema un ingresso a rampa unitaria e si può vedere un errore che oscilla tra 0 e 0.07 con una media di circa 0.035, può considerarsi soddisfatta la specifica imposta sull'errore per gli ingressi a rampa minore uguale di 0.04.

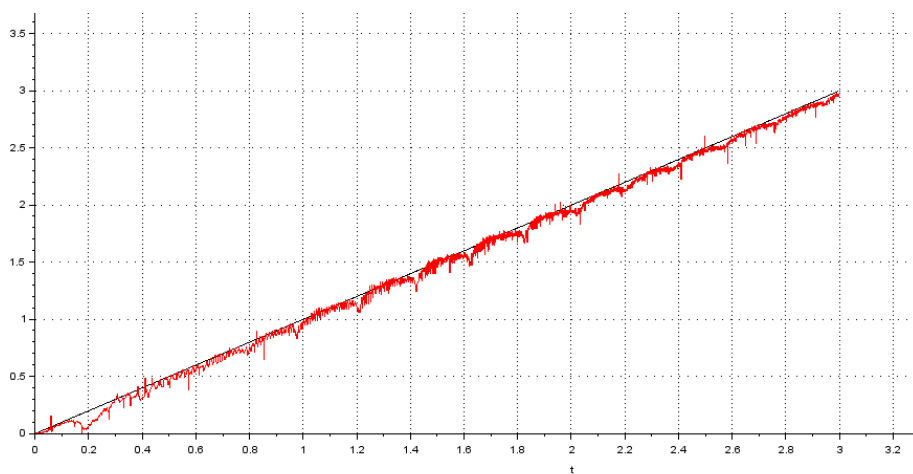


Figura 41: Risposta del sistema retroazionato a un ingresso a rampa unitaria.

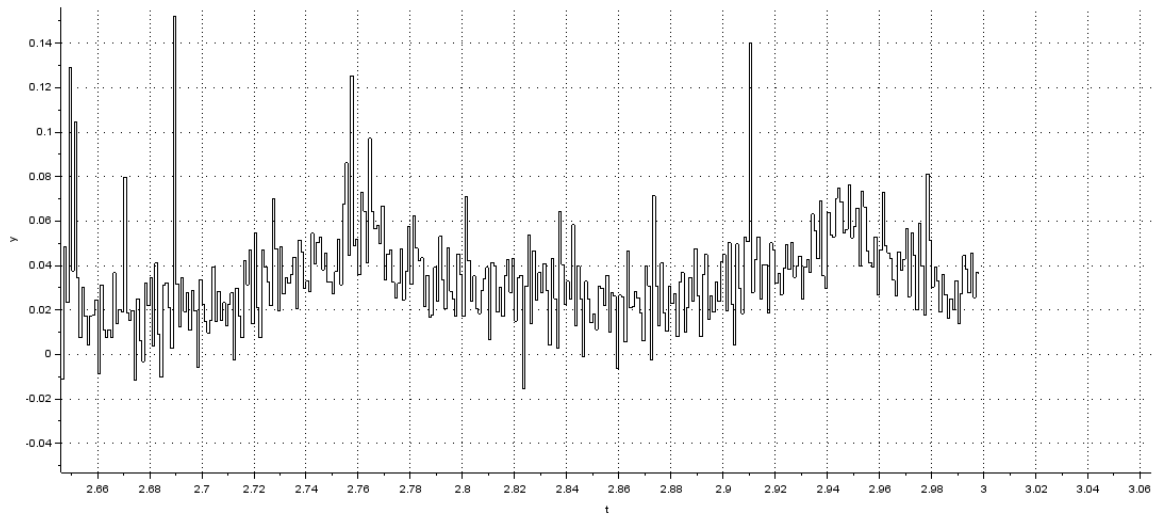


Figura 42: Errore tra ingresso e uscita desiderata per la rampa unitaria.

Viene data in ingresso al sistema anche una sinusoide, di frequenza 0.5 rad/s, la risposta del sistema segue discretamente l'ingresso.

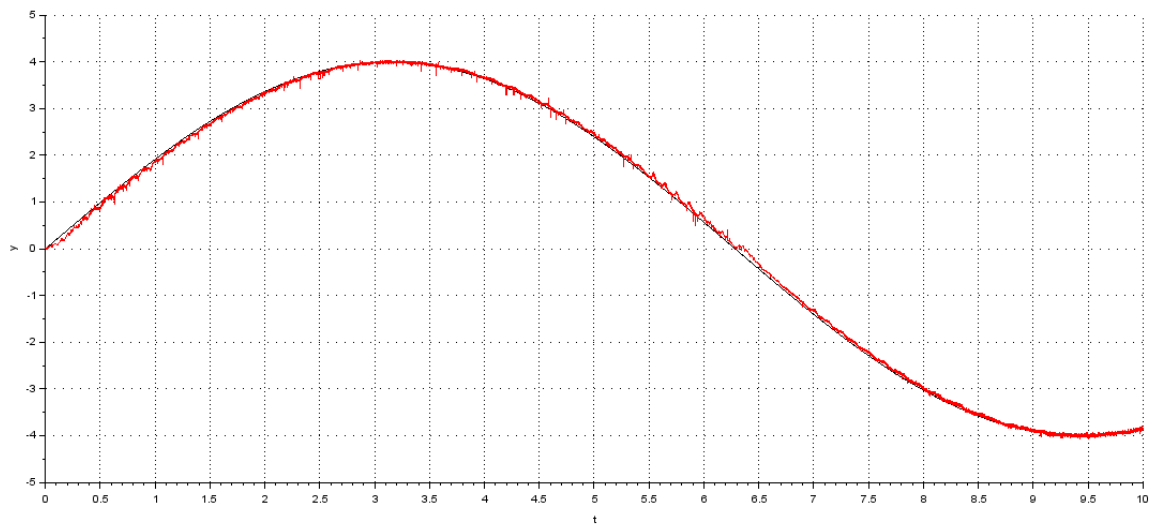


Figura 43: Risposta del sistema retroazionato a un ingresso sinusoidale da 0.5 rad/s.

Si potrebbe aumentare il guadagno k per migliorare la risposta sia alla rampa che alla senoide ma ciò porterebbe un peggioramento nella risposta al gradino con una maggiore sovraelongazione.

Esempio di risposta con $k=27$ e $k=28$

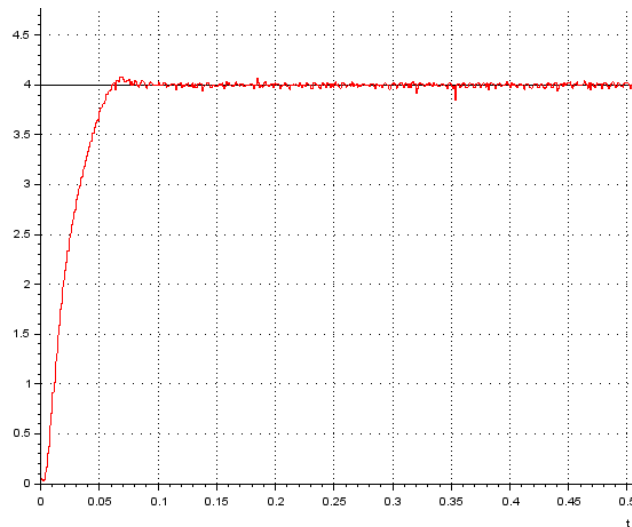


Figura 44: Risposta del sistema retroazionato a un ingresso a gradino da 4V con $k=27$.

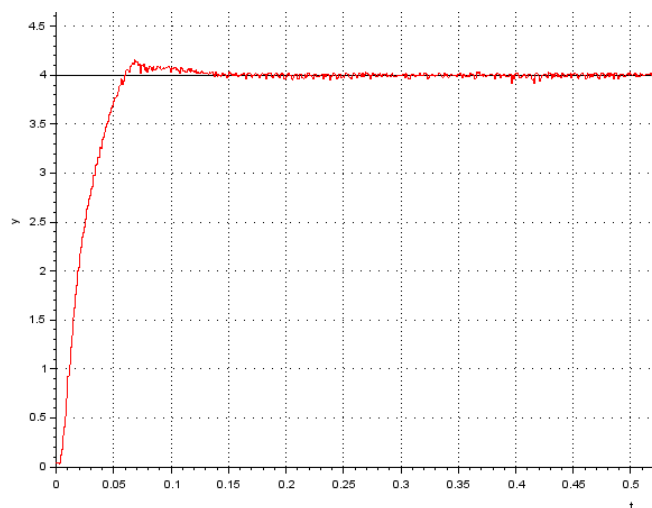


Figura 45: Risposta del sistema retroazionato a un ingresso a gradino da 4V con $k=28$.

Si può notare come all'aumentare del guadagno k aumenta la sovravelongazione della risposta al gradino.

4.3 Controllore progettato con il luogo delle radici

Viene testato il controllore progettato con il luogo delle radici

$$G(s) = \frac{26 \frac{s}{30} + 1}{s \frac{s}{90} + 1}$$

Viene dato in ingresso al sistema un gradino di ampiezza 4.

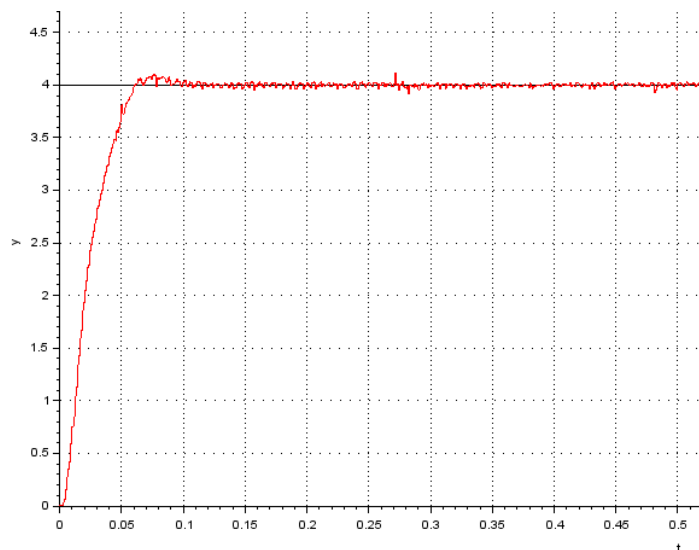


Figura 46: Risposta del sistema retroazionato a un ingresso a gradino da 4V.

Il tempo di salita è di circa 0.04 s, minore di 0.07 s, ma si nota una sovravelongazione non indifferente, circa del 2.5%.

Un modo per ridurre tale sovralongazione è di ridurre il guadagno k o spostare più a sinistra il polo del controllore, il sistema infatti ha un transitorio più stabile se il polo è più distante dallo zero. Non conviene abbassare il guadagno k che porterebbe risposte peggiori agli ingressi a rampa e agli ingressi sinusoidali e perciò provo a spostare il polo del controllore più lontano dallo zero e dall'asse immaginario.

Provo quindi il controllore

$$G(s) \frac{26 \frac{s}{30} + 1}{s \frac{s}{100} + 1},$$

con il polo della funzione anticipatrice a 100 rad/s invece che a 90 rad/s.

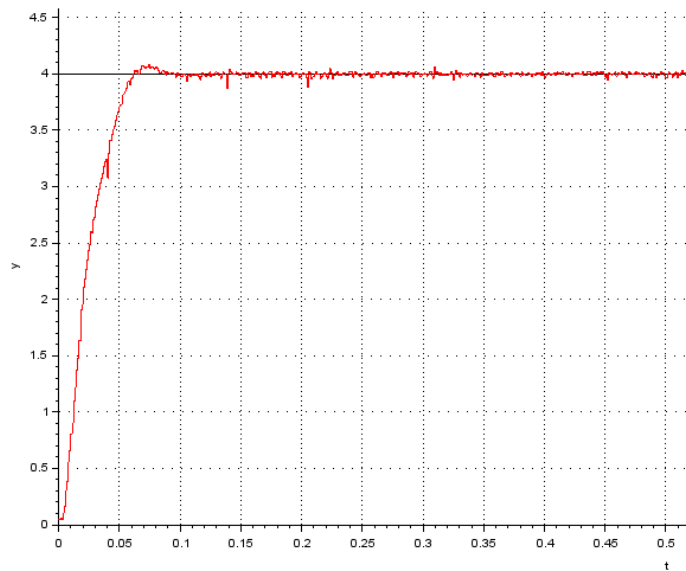


Figura 47: Risposta del sistema retroazionato a un ingresso a gradino da 4V.

Si può notare un transitorio migliore, con una sovralongazione circa del 1.25%.

Se si spostasse il polo ulteriormente a sinistra la sovraelongazione diminuirebbe ancora, per esempio con il polo a 110 rad/s la sovraelongazione è circa 0.5%.

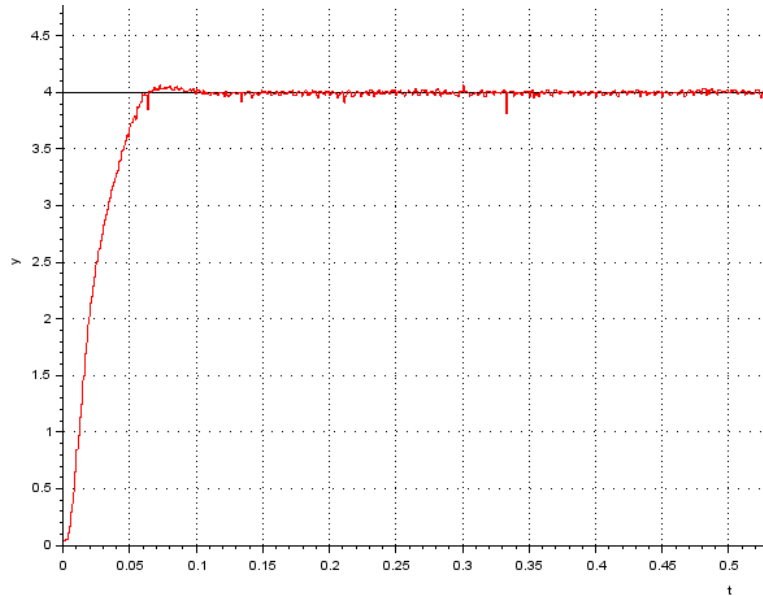


Figura 48: Risposta del sistema retroazionato a un ingresso a gradino da 4V.

Ora si testa il controllore con un ingresso a rampa unitaria.

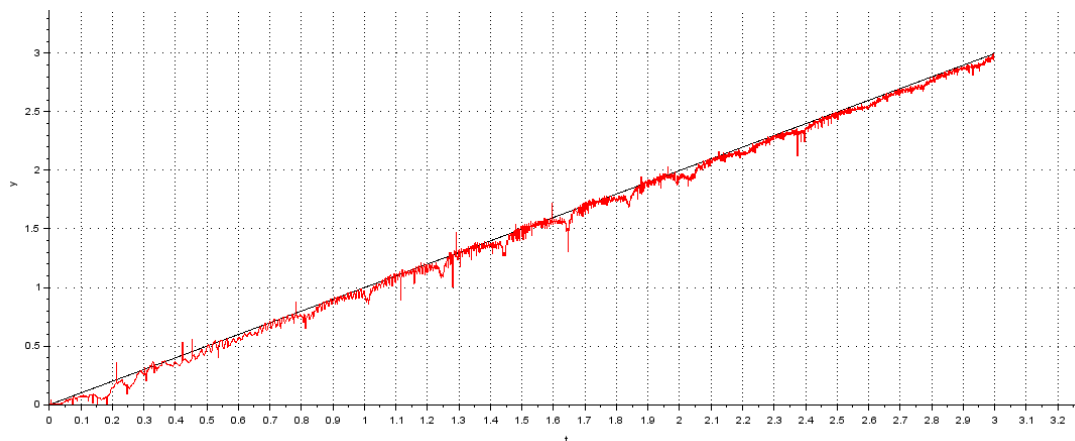


Figura 49: Risposta del sistema retroazionato a un ingresso a rampa unitaria.

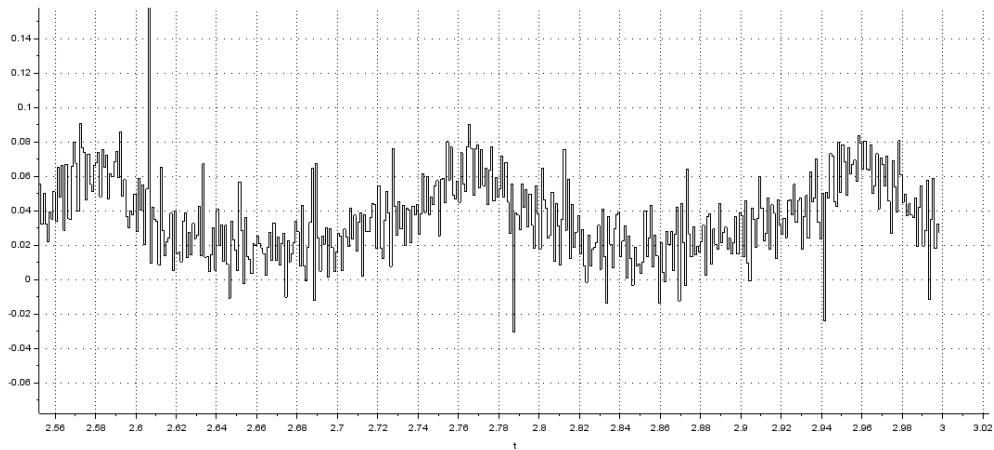


Figura 50: Errore tra ingresso e uscita desiderata per la rampa unitaria.

Viene data in ingresso al sistema un ingresso a rampa unitaria e c'è un errore che oscilla tra 0 e 0.07 con una media di circa 0.035, può considerarsi soddisfatta la specifica imposta sull'errore per gli ingressi a rampa minore uguale di 0.04.

Viene data in ingresso al sistema anche una sinusoidale, di frequenza 0.5 rad/s, la risposta del sistema segue discretamente l'ingresso.

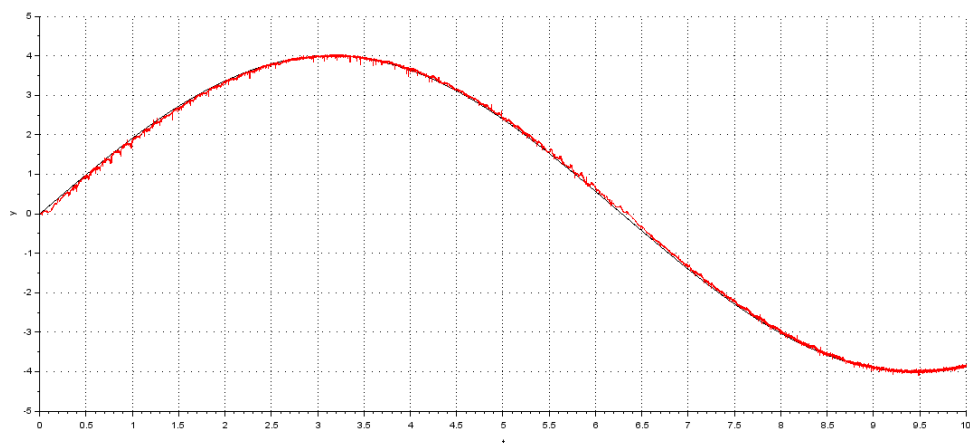


Figura 51: Risposta del sistema retroazionato a un ingresso sinusoidale da 0.5 rad/s.

Non posso aumentare il guadagno k per migliorare la risposta sia alla rampa che alla sinusoidale perché ciò porterebbe un peggioramento nella risposta al gradino con una maggiore sovraelongazione.

Conclusioni

Le discrepanze tra le risposte del modello simulato e del sistema reale potrebbero essere dovute a imprecisioni e semplificazioni nella stima del processo. È stata fatta un'identificazione del sistema presupponendo come modello teorico un sistema del primo ordine quando in realtà il motore ha due poli, quello elettrico e quello meccanico, per quanto il polo elettrico è di gran lunga maggiore del polo meccanico e in prima approssimazione può essere trascurato. Anche le altre componenti hardware che compongono il sistema potrebbero necessitare di considerazioni che sono state trascurate.

È stato necessario, perciò, adattare in corso d'opera il controllore sintetizzato con il luogo delle radici. ma una volta fatto ciò è risultato soddisfacente. Fortunatamente poi la scheda MicroDAQ permette di testare velocemente le applicazioni progettate e questo permette di verificare in tempo reale la qualità dei controllori sviluppati.

L'obiettivo di studiare e sviluppare un'applicazione per il controllo e l'acquisizione dati in real-time per DSP risulta quindi raggiunto con successo ed entrambe le applicazioni sviluppate sono soddisfacenti. Vengono raccolti tutti i dati necessari all'identificazione del processo in maniera semplice e veloce, e anche l'applicazione di controllo funziona in maniera efficiente.

Bibliografia e sitografia

- [1] P. Bolzern, R. Scattolini, N. Schiavoni, Fondamenti di controlli automatici, McGraw-Hill, 2015.
- [2] G. F. Franklin, J. D. Powell, A. Enami-Naeini, Controllo a retroazione di sistemi dinamici Vol 1, ediSES, 2004.
- [3] L. Tan, *Digital Signal Processing: Fundamentals and Applications*, Academic Press, 2007.
- [4] S. W. Smith, *The Scientist and Engineer's Guide to Digital Signal Processing*, California Technical Publishing , 1999.
- [5] MicroDAQ, <http://www.microdaq.org/>.
- [6] Scilab, <https://www.scilab.org/>.
- [7] Matlab, <https://it.mathworks.com/products/matlab.html>.