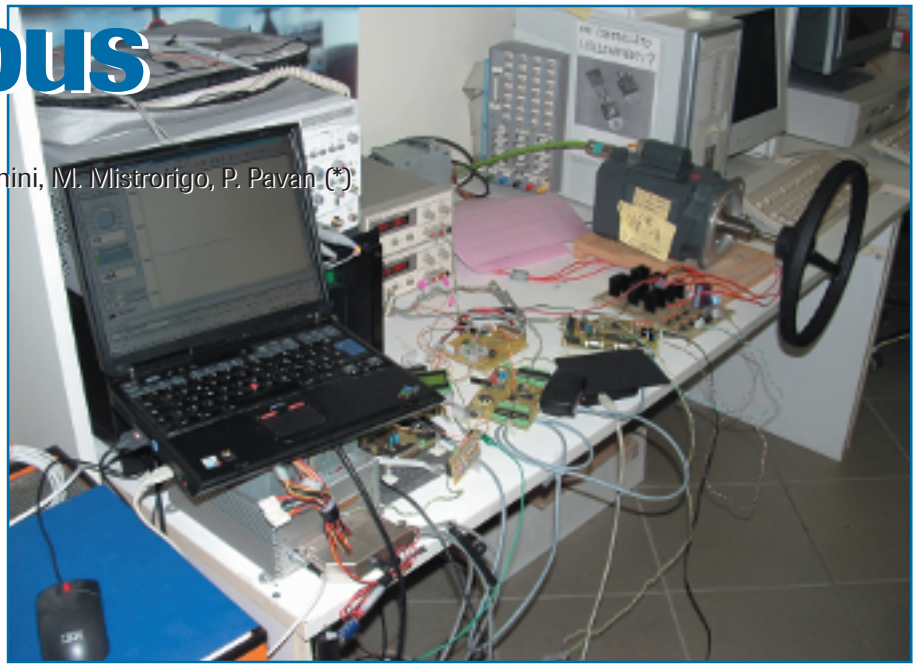
Sperimentando con CANbus

A. Bertacchini, L. Tamagnini, M. Mistrorigo, P. Pavan (*)



Il banco di prova utilizzato per l'esperimento

Controllo di un motore brushless con ridondanza statica per applicazioni di force feedback in sistemi steer-by-wire

In questo articolo viene proposto un approccio Hardware-in-the-loop (HIL) per la realizzazione di un'architettura preliminare di controllo di force feedback in applicazioni Steer-by-wire (SBW).
Come attuatore è stato scelto un motore brushless.
La determinazione della posizione dell'albero motore gioca un ruolo chiave nell'algoritmo di controllo.
Per ottenere una posizione affidabile dell'albero motore si è scelto di implementare una classica ridondanza statica tripla (TMR - Triple Modular Redundancy).
I segnali provenienti dall'encoder integrato nel motore vengono elaborati sfruttando tre piattaforme hardware: la prima si basa su un microcontrollore a 8-bit; la seconda su un microcontrollore a 16-bit; la terza sfrutta un modulo software integrato nel tool di sviluppo utilizzato come hardware virtuale. Quest'ultima è impiegata anche come voter.
La posizione effettiva dell'albero motore, risultato dell'algoritmo di voting, è inviata via bus CAN alla piattaforma a 16-bit, la quale implementa l'algoritmo di controllo del motore e genera i segnali PWM necessari all'alimentazione dello stesso.
La comunicazione tra hardware virtuale e reale avviene attraverso il bus CAN.
Esperimenti condotti con diverso baudrate confermano la validità dell'algoritmo di voting implementato, che produce risultati corretti anche in caso di guasto in uno dei moduli dell'architettura TMR e non è influenzato da condizioni di alto carico del bus di comunicazione.

La guida assistita

Cos'è un sistema Steer-by-wire (SBW)? È un sistema di guida assistita che richiede un controllo elettronico diretto dello sterzo e sostituisce il tradizionale back-up meccanico o idraulico con un sistema meccatronico distribuito e fault-tolerant.
Un sistema SBW completo porta diversi vantaggi in termini di consumi di carburante e pneumatici, migliora le funzioni ergonomiche e il comfort di guida e soprattutto la sicurezza del guidatore e dei passeggeri, grazie alla presenza di apparati elettronici distribuiti fault tolerant che controllano i comandi di sterzo. Inoltre, grazie all'assenza del piantone dello sterzo è possibile realizzare abitacoli più sicuri in grado di superare crash test molto severi.
Naturalmente i benefici introdotti dalla tecnologia 'by-wire' sono maggiori se in un veicolo vi è integrazione fra più sistemi x-by-wire, per esempio sterzo, freni, acceleratore, sospensioni attive ecc…
Si può pensare a un sistema SBW (fig.1) come com-

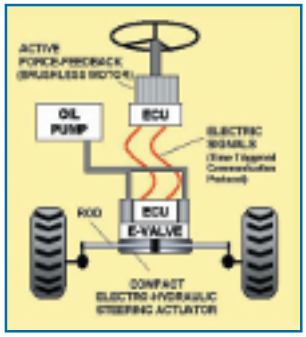


Fig.1: Lo schema illustra l'architettura di un sistema steer-by-wire

posto di 3 sottosistemi: la parte di azionamento del comando di sterzo, costituita da una centralina a microcontrollore che controlla un cilindro idraulico per mezzo di un'elettrovalvola; il sistema di force feedback, che consta anch'esso di una centralina a microcontrollore o Fpga, volante (o joystick) e attuatore di force feedback (che per noi è un motore brushless); rete di comunicazione tra le centraline.
In sistemi by-wire è richiesto l'uso di protocolli dal comportamento deterministico (time-triggered). Al momento non è ancora stato definito uno standard, per cui i test sui nostri prototipi sono stati effettuati adottando il classico protocollo CAN event-triggered, quindi intrinsecamente non deterministico. In questo particolare caso, però, gli svantaggi derivanti dall'uso di un simile protocollo sono stati ridotti grazie al fatto che è stata implementata una rete dedicata. Di seguito viene descritto un approccio HIL per l'implementazione di un sistema di force feedback per un'applicazione SBW generica. In particolare l'attenzione è posta sui caratteri di fault tolerance e di autodiagnostica del sistema proposto.

Tecniche di fault tolerance

I sistemi fault-tolerant sono in grado di continuare a funzionare normalmente anche in caso di guasti hardware o software. Nel progetto di un sistema fault tolerant, quindi, devono essere tenuti in considerazione aspetti di 'error processing', ossia rimozione di errori computazionali prima che il guasto si verifichi, e 'fault containment', ossia un guasto non deve propagarsi ad altri moduli del sistema.
Per implementare sistemi fault-tolerant è necessario utilizzare ridondanze (allo specifico modulo vengono aggiunti, solitamente in parallelo, uno o più moduli equivalenti). I concetti di ridondanza riguardano, in generale, l'hardware, il software, i dati scambiati e i riferimenti temporali di sincronizzazione. Il nostro sistema propone una ridondanza hardware, per la quale esistono tre possibili forme d'implementazione: passiva, attiva e ibrida.
Con una ridondanza passiva (o statica) si raggiunge la fault tolerance senza eseguire alcuna azione. Nella versione più semplice non esiste un riconoscimento dell'errore e il guasto è mascherato staticamente dal voter, il modulo di scelta del dato effettivo, indispensabile in ogni architettura ridondante.
La ridondanza attiva, invece, è una tecnica dinamica che richiede meno moduli hardware al prezzo, però, di un aumento della complessità della struttura del voter e degli algoritmi decisionali in esso implementati.
Normalmente questa tecnica prevede meccanismi d'individuazione dei guasti e, soprattutto, una riconfigurazione del sistema per escludere il nodo in cui si è verificato l'errore. Un approccio di tipo ibrido, infine, combina i precedenti due.

L'architettura ridondante

In fig.2 si può vedere il diagramma a blocchi semplificato dell'architettura proposta, di tipo TMR statica (passiva), applicata al sottosistema di determinazione della posizione angolare del rotore di un motore brushless, impiegato come attuatore per il force feedback. Con quest'architettura è tollerato un solo modulo difettoso alla volta. Il guasto può essere funzionale o tecnologico, singolo o multiplo, ma deve rimanere confinato all'interno del modulo (requisito di fault containment) ed è mascherato dall'algoritmo implementato nel voter.
Da un punto di vista hardware un'architettura TMR può essere applicata a differenti livelli: di applicazione, triplicando l'intero sistema, oppure di sistema, triplicando il processore, come in questo caso. I tre moduli sono stati implementati adottando tre diverse piattaforme, ognuno con differenti caratteristiche. Di conseguenza, sono state utilizzate tre strategie di determinazione della posizione angolare, quindi implicitamente sono stati implementati anche alcuni aspetti di ridondanza software, nonché un semplice algoritmo di autodiagnostica del sistema.

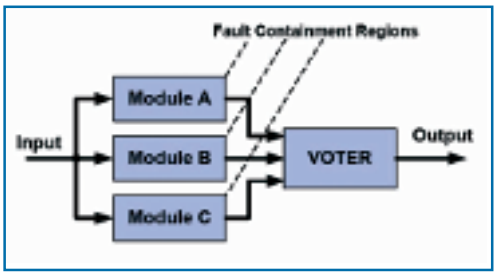


Fig.2: Diagramma a blocchi semplificato dell'architettura ridondante TMR (Triple Modular Redundancy)

Il sistema di force feedback

Il sistema proposto si basa su un approccio Hardware-in-the-loop (HIL), che si riferisce a una metodologia di sviluppo in cui, partendo da un modello del sistema da realizzare (hardware virtuale) si sostituiscono, in più fasi successive, i vari componenti virtuali del sistema stesso con gli equivalenti componenti reali, per ottenere un sistema finale reale equivalente al modello iniziale.
Questa tecnica permette di testare le unità di controllo elettroniche (ECU) in maniera sistematica e in gran parte in ambiente virtuale, quindi in completa sicurezza.
Inoltre, permette di ottimizzare hardware e software nelle fasi iniziali del progetto, comportando a lungo termine una certa riduzione dei costi.
Un diagramma a blocchi semplificato del sistema di force feedback realizzato è mostrato in fig.3.
Sull'albero motore del motore brushless usato come

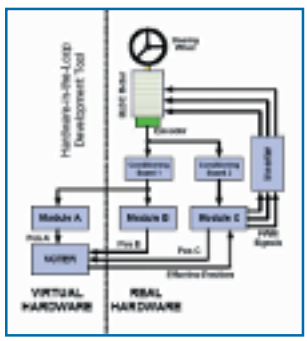


Fig.3: Schema a blocchi semplificato del sistema di force feedback

attuatore di force feedback è stato calettato direttamente un volante. Il motore ha un encoder integrato all'interno i cui segnali, opportunamente condizionati, sono utilizzati dai tre moduli dell'architettura TMR (A, B e C) per stimare la posizione dell'albero motore (volante).
Il primo modulo dell'architettura TMR (A) è un componente virtuale e il suo comportamento è stato realizzato sfruttando le potenzialità di un tool di sviluppo HIL commerciale.
Il secondo modulo (B), invece, è reale e stima la posizione del rotore attraverso un microcontrollore a 8-bit. Anche il terzo modulo (C) è reale e determina la posizione del rotore utilizzando un microcontrollore a 16-bit. I tre valori di posizione stimati sono inviati al voter (hardware virtuale) che, sulla base della strategia di voting implementata, decide la posizione effettiva dell'albero motore e la invia al modulo C.
Oltre alla determinazione della posizione del volante, quest'ultimo esegue anche il controllo di coppia del motore, generando i segnali PWM necessari a comandare l'inverter che alimenta e controlla fisicamente il motore.
Tutto l'hardware reale utilizzato nel sistema è stato progettato e realizzato nei laboratori dell'Università.

Algoritmo di voting

I valori di posizione stimati dai tre moduli possono essere diversi, entro un determinato intervallo, anche in assenza di malfunzionamenti. Per questa ragione, non è possibile utilizzare una classica strategia di 'Majority voting' e si è deciso di implementare nel voter una strategia MVS (Mid-Value Select). I valori di posizione stimati dai tre moduli sono ordinati in tempo reale e ne viene estratto il mediano, il quale corrisponde alla posizione effettiva inviata al modulo C e usata da quest'ultimo per eseguire il controllo di coppia sul motore. In questo modo, in caso di guasto in uno dei tre moduli, il corrispondente dato di posizione è automaticamente scartato, perché nel vettore ordinato dei valori andrà ad occupare la posizione più alta o più bassa.
Il voter implementa anche un semplice processo di autodiagnostica. In particolare, esso controlla se i valori di posizione provenienti dai moduli dell'architettura ridondante rientrano in un range dinamico dell'ultimo valore di posizione calcolato dal voter stesso. Se un segnale esce da tale range per un certo numero di volte consecutive, allora il nodo è classificato come guasto e il voter invia sul bus un messaggio contenente il relativo codice di errore, per notificare la condizione di malfunzionamento. Ogni codice è univoco e generato con una semplice funzione logica combinatoria.
Una volta che un nodo è classificato come corrotto, il voter cambia la strategia di voting e viene presa come posizione effettiva la media dei valori dei due moduli rimanenti.

Il ruolo della rete CAN

I dati sono scambiati tra hardware reale e virtuale usando una classica comunicazione su bus CAN. Nella semplice rete implementata compare anche un nodo di monitor del sistema, realizzato in questo caso da un analizzatore di reti CAN commerciale. Il voter che implementa un semplice processo di autodiagnostica, invia sul bus un messaggio che riferisce lo stato dell'intero sistema. Nel prototipo realizzato questo messaggio viene ricevuto dal nodo di monitor, ma in un sistema completo, ad esempio su un veicolo, questo messaggio potrà essere ricevuto da altre ECU che, in caso di malfunzionamento, potranno portare il sistema in condizioni di sicurezza. Inoltre, la posizione stimata dall'architettura TMR descritta può essere utilizzata da altre ECU per migliorare le prestazioni e la sicurezza del veicolo.

Risultati sperimentali

Per validare il sistema sono stati fatti esperimenti HIL. In particolare, sono state eseguite prove di tracking; per questo, però, è stato necessario fornire al sistema due parametri aggiuntivi (fig.4): la posizione delle ruote e la rigidità dello sterzo (steering stiffness, un parametro elettronico che riproduce in un sistema SBW la rigidità del piantone dello sterzo presente nei sistemi di guida tradizionali).
Questi due segnali sono generati dalla piattaforma di sviluppo di hardware virtuale e sono inviati al modulo C via bus CAN.
Il modulo C utilizza questi dati e la posizione effettiva del volante ricevuta dal voter per eseguire un controllo di coppia del motore, basato sulla differenza tra posizione ruote (virtuale) e posizione volante (reale). Eseguendo gli stessi test con differente baudrate si è verificato come il

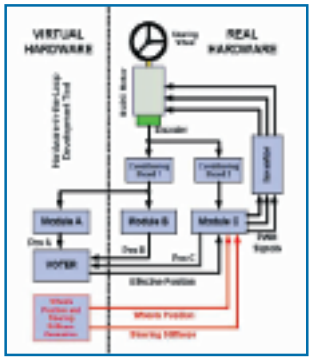


Fig.4: Riferimento sulla posizione delle ruote e rigidità del piantone dello sterzo (steering stiffness) forniti al sistema dalla piattaforma di hardware virtuale

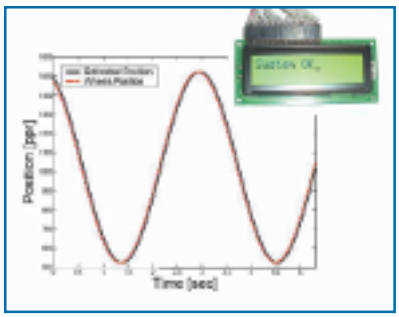
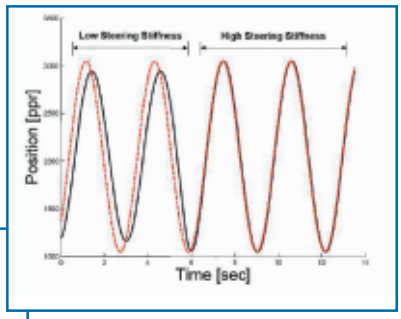


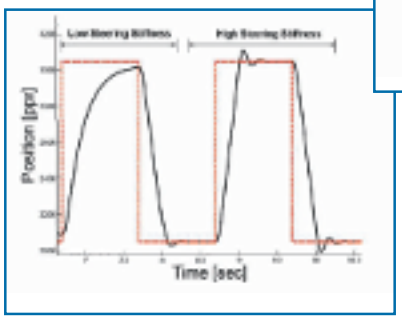
Fig.5: Tracking test: riferimento di posizione delle ruote (rosso) e posizione del volante stimata dal sistema (nero)

sistema funzioni correttamente anche in caso di alto carico del bus. I risultati riportati di seguito si riferiscono a trasmissioni CAN a 1 Mbaud.
La fig.5 mostra il comportamento del sistema quando un'onda sinusoidale fa da riferimento della posizione delle ruote (linea rossa); da un punto di vista pratico essa corrisponde a una serie di continue sterzate destra-sinistra delle ruote.
È possibile vedere come la posizione del volante (linea nera) insegua perfettamente il riferimento; ciò significa che l'algoritmo di controllo di coppia, la strategia di voting e le strategie di determinazione della posizione del volante implementate nei singoli moduli funzionano correttamente.
Cambiando i parametri, ad esempio la rigidità dello sterzo, muta anche il comportamento generale del sistema (fig.6A). In particolare, con una bassa rigidità dello sterzo il sistema è poco reattivo e introduce alcuni ritardi nell'inseguimento del riferimento; viceversa con un'alta rigidità il sistema è più reattivo, ma introduce overshoot dovuti, ad esempio, all'inerzia meccanica del volante, i quali possono tradursi in vibrazioni sul volante.

Fig.6 (A+B): Effetti di variazione della rigidità dello sterzo, con riferimento di posizione delle ruote rispettivamente sinusoidale (rosso-fig.A) e a gradino (rosso-fig.B) e posizione del volante stimata dal sistema (nero)



A



B

Gli effetti della variazione della rigidità sono più evidenti se viene utilizzata come riferimento di posizione delle ruote un'onda quadra (fig.6B), la quale da un punto di vista pratico corrisponde a brusche sterzate destra-sinistra delle ruote. Si possono migliorare le prestazioni utilizzando algoritmi di controllo di coppia più complessi, come un controllo in anello chiuso sulle correnti di fase del motore.
In caso di guasto in uno dei moduli il sistema continua a funzionare correttamente, a riprova della validità delle soluzioni adottate.
In fig.7, infatti, è possibile notare come, nonostante il guasto indotto nel modulo B, la posizione del volante continui a seguire il riferimento delle ruote. Grazie al processo di autodiagnostica implementato il voter rileva il guasto, per cui il messaggio di stato del sistema cambia passando da 'System ok'

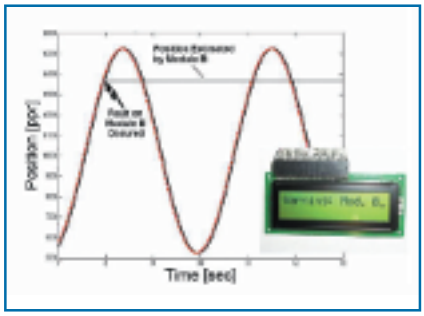


Fig.7: Induzione di guasto sul modulo B: l'architettura ridondante maschera il guasto e il sistema continua a funzionare correttamente

a 'Warning on Mod.B'; contemporaneamente cambia la strategia di decisione che passa da 'mid value' a 'mean value'; il sistema, però, continua a funzionare correttamente.

Tiriamo le somme

I sistemi SBW rientrano nella categoria di applicazioni classificate come 'safety critical' e richiedono l'uso di protocolli di comunicazione dal comportamento deterministico (time-triggered). Da un punto di vista logico, quindi, non sarebbe corretto utilizzare un protocollo event-triggered non deterministico come CAN, ma in questo caso particolare è stata implementata una rete dedicata, per cui gli svantaggi legati all'uso di un protocollo event-triggered si sono considerevolmente ridotti.
Obiettivi principali di questo lavoro erano: verificare le strategie di determinazione della posizione angolare del rotore del motore, quindi del volante, utilizzando architetture a 8 e 16 bit; conseguentemente verificare l'algoritmo di controllo di coppia del motore; verificare l'architettura ridondante proposta e l'algoritmo di voting implementato dal voter.
I risultati sperimentali mostrano come questi obiettivi siano stati raggiunti e come il sistema si comporti correttamente anche nel caso in cui un modulo presenti malfunzionamenti. Questo significa che l'architettura TMR proposta può essere impiegata come punto di partenza per implementare anche una completa ridondanza software e un'architettura ridondante più complessa.
Sviluppi futuri del sistema prevedono l'implementazione di una ridondanza estesa anche al sottosistema 'voting' e il passaggio a un protocollo time-triggered una volta che questo sia standardizzato. ■

(*) Università degli Studi di Modena e Reggio Emilia, studio presentato in occasione di Icc, International CAN Conference (CAN in Automation)

Università degli Studi di Modena e Reggio Emilia
Readerservice.it n. 55