Università degli Studi di Padova

FACOLTÀ DI INGEGNERIA
Corso di Laurea in Ingegneria dell'Informazione

Tesi di laurea triennale

Networked Control System

Candidato:
Guidolin Arianna

Relatore:
Schenato Luca

Anno Accademico 2012–2013

Indice

Sommario

La tradizionale connessione point-to-point, utilizzata per decenni nell'industria, è ormai stata soppiantata dalle *Networked Control Systems*, grazie ai numerosi vantaggi che offrono, quali la facilità di installazione e manutenzione, l'ampia flessibilità e i costi contenuti. Inoltre, inserire un sistema di controllo su rete permette di progettare metodi in grado di aumentare le prestazioni della trasmissione, come il controllore *Packet-Based*, capace di spedire più valori utili di controllo in un singolo pacchetto.

In questa tesi, oltre ad una descrizione teorica dei parametri di rete e di controllo delle *Networked Control Systems*, viene simulata una rete con l'utilizzo di MATLAB, per poter analizzare i vantaggi offerti da questo tipo di connessione.

Introduzione

Un sistema di controllo retroazionato, dove gli anelli di controllo sono chiusi attraverso una rete, viene detto *Networked Control System* (NCS). In molti sistemi complicati, come veicoli, impianti industriali o smart house, l'uso delle NCSs viene ampliamente utilizzato perché, rispetto alla più tradizionale connessione point-to point, ha molti vantaggi tra i quali il basso costo, la riduzione di cavi e potenza richiesta, nonchè una bassa manuntezione ed elevata affidabilità. L'inserimento di una rete di communicazione in un anello di controllo retroazionato rende l'analisi e la progettazione di una NCS complessa. I principali problemi da affrontare sono i ritardi introdotti dalla rete (ritardo tra sensori e controllore e ritardo tra controllore e attuatore) e la possibilità che alcuni pacchetti possano essere persi durante la trasmissione.

Le reti sono progettate a seconda della loro funzione, le principali sono: controllo, diagnosi e sicurezza. Le operazioni di controllo generalmente si riferiscono alla comunicazione tra sensori e attuatori attraverso il controllore. Mentre le reti atte alla diagnosi, si riferiscono a sistemi in grado di trasmettere una consistente quantità di dati non frequentemente. Infine le reti di sicurezza permettono ai sistemi di essere facilmente gestiti e rinconfigurati in caso di errori. Ognuna di queste reti necessita di particolari caratteristiche, ad esempio è importante un elevato determinismo per il controllo e la sicurezza, mentre per la diagnosi è necessaria una rete in grado di gestire pacchetti con elevate quantità di dati da trasmettere velocemente. Per questo la scelta del tipo di protocollo utilizzato è molto importante. Il protocollo, infatti, descrive il comportamento di una rete secondo i modelli ISO/OSI e il funzionamento dei sette livelli di rete: fisico, data link, network, trasporto, sessione, presentazione e applicazione.

In questa tesi vengono descritti i principali parametri di una rete, che includono tempo di ritardo e propagazione, lunghezza di pacchetto e il MAC, *Media Access Control*, dei principali protocolli utilizzati in una Networked Control System: Ethernet, ControlNet e DeviceNet. In seguito vengono analizzate la progettazione delle strategie di controllo del sistema retroazionato, con particolare attenzione al ritardo e alla perdita di pacchetti. Infine viene presentata una simulazione di rete con MATLAB e se ne analizza l'andamento qualora avvenga una perdita di pacchetti.

Capitolo 1

Control Network

La funzione di una rete è di trasmettere dati da un nodo all'altro. Ma ogni rete può avere caratteristiche molto differenti e la scelta su come organizzarle dipende dal tipo di applicazione che si vuole implementare e dalle prestazioni che si vogliono ottenere.

Il funzionamento di una rete è descritto dal MAC (*Media Access Control*), che ne determina tutte le caratteristiche, come la divisione dell'ampiezza di banda, la risoluzione delle contese tra i vari nodi o la lunghezza di pacchetto. In questa sezione vengono descritti i MAC di tre tipi di Control Networks molto diffusi: Ethernet, ControlNet e DeviceNet.

1.1 Ethernet

Ethernet usa il meccanismo CSMA/CD (*Carrier Sense Multiple Access with Collision Detection*) per risolvere le contese nella trasmissione. Secondo il protocollo CSMA/CD, quando un nodo vuole trasmettere prima ascolta la rete e se questa è occupata attende che sia libera, altrimenti trasmette immediatamente. Se due o più nodi trovano la rete libera e decidono di trasmettere simultaneamente avviene una collisione. Durante la trasmissione, i nodi sono in grado di rilevare se il messaggio che stanno inviando è corrotto a causa di una collisione. In questo caso i nodi interrompono la trasmissione e attendono prima di riprovare nuovamente. Il tempo di attesa è determinato dall'algoritmo *Standard Binary Exponential Backoff* (BEB): la successiva ritrasmissione avviene in un tempo scelto in modo casuale tra 0 e $(2^i - 1)$ tempi di slot, dove i denota l'i-esima collisione avvenuta e il tempo di slot è il tempo minimo necessario per un intero ciclo di trasmissione. Dopo 16 trasmissioni fallite il nodo sospende i tentativi e comunica ai layer più alti l'insuccesso. La struttura dei pacchetti è mostrata in Figura 1.1: consiste in un overehead di 26 bytes, mentre i dati da trasmettere hanno una dimensione dai 46 a 1500 bytes.

Il principale vantaggio dell'Ethernet per reti non sovraccariche sono i bassi tempi di ritardo; inoltre non viene utilizzata larghezza di banda per ottenere l'accesso alla rete, a differenza dei protocolli che si basano sul token bus o token ring. Tuttavia Ethernet non è un protocollo deterministico e non prevede l'uso di priorità per alcuni messaggi. Ciononstante il problema maggiore di questo protocollo sono le frequenti collisioni che avvengono quando la rete è carica.

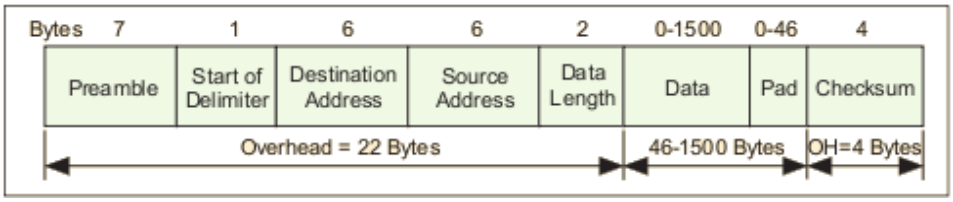

Figura 1.1: Struttura del pacchetto Ethernet.

1.2 ControlNet

ControlNet, come pure MAP e PROFIBUS, sono tipici esempi di reti con protocollo token-passing bus. Sono reti deterministiche perché il massimo tempo di attesa prima di spedire un pacchetto è determinato dal tempo di rotazione del token. I nodi in una rete token bus sono disposti logicamente ad anello e, nel caso del ControlNet, ogni nodo conosce l'indirizzo del suo predecessore e del suo successore. Nella rete, il nodo con il token può trasmettere pacchetti finchè non li ha esauriti o la durata del tempo in cui ha tenuto il token ha raggiunto il limite. In seguito, il nodo rigenera il token e lo trasmette al suo successore logico nella rete. Se il nodo non ha messaggi da inviare, allora semplicemente passa il token al nodo sucessivo. La locazione fisica dei nodi non è importante perché i token sono trasmessi secondo una struttura logica.

I pacchetti non collidono mai perché solo un nodo può trasmettere alla volta. La stuttura dei pacchetti è indicata in Figura 1.2: sono formati da un overhead di 7 bytes, mentre i dati hanno una dimensione compresa dai 0 ai 510 bytes.

Rispetto alla rete Ethernet, il protocollo token bus è deterministico e ha alti livelli di efficienza per reti trafficate, mentre per reti con pochi pacchetti da trasmettere le prestazioni sono inferiori. Un'altra importante caratteristica del token bus è che può dinamicamente aggiungere o rimuovere nodi. Questo non avviene nel caso del token ring, dove i nodi formano un anello logico che non può essere modificato.

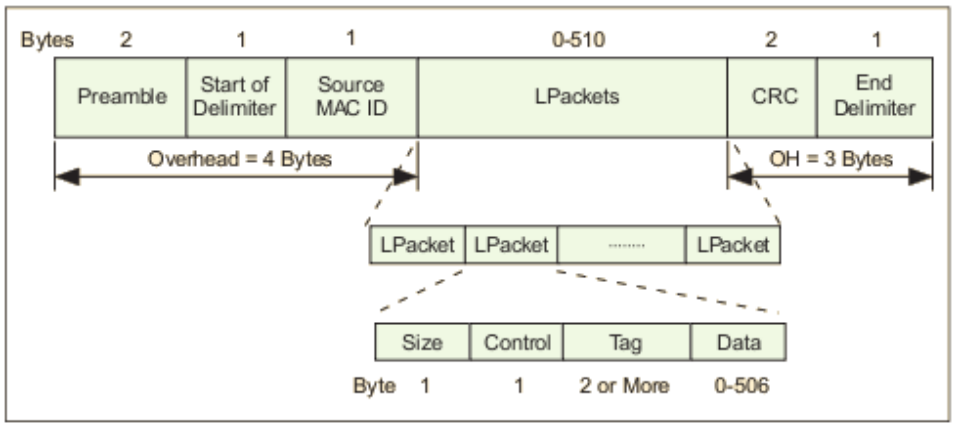

Figura 1.2: Struttura del pacchetto ControlNet.

1.3 DeviceNet

DeviceNet utilizza il protocollo CAN (*Controller Area Network*), è un protocollo utilizzato per molte applicazioni nell'industria automobilistica, ma in generale è sfruttato dove il tempo è un fattore cruciale. Il protocollo è ottimizzato per l'invio di messaggi corti e utilizza CSMA/AMP (*CSMA/Arbitration on Message Priority*): tutti i messaggi hanno una specifica priorità che determina l'accesso alla rete nel caso di trasmissioni simultanee. Un nodo che vuole trasmettere un messaggio attende finchè la rete è libera e poi comincia ad inviare l'identificatore del suo messaggio bit per bit. L'identificatore determina la priorità: lo zero è dominante rispetto all'uno. Quindi se due nodi cercano di trasmettere nello stesso momento continuano ad inviare bit e ascoltano la rete. Se uno di loro riconosce un bit differente da quello che ha inviato, allora perde il diritto di trasmettere e l'altro nodo, che ha avuto la priorità, continua ad inviare il messaggio. In questo modo, la collisione non altera la trasmissione di entrambi i nodi.

Nelle reti basate sul protocollo CAN i dati sono trasmessi da un nodo ad un altro, oppure a più nodi. Infatti, nei pacchetti non è presente necessariamente l'indirizzo del destinatario, ma sono contrassegnati da un'etichetta univoca che serve ai nodi che ricevono il messaggio per determinare se accettarlo o meno. In Figura 1.3 è mostrato il formato del pacchetto: 47 bits sono utilizzati per l'overhead, mentre i dati sono compresi dai 0 ai 8 bytes.

DeviceNet è diffusa in varie applicazioni industriali perché è una rete a basso costo ed è ideale per la trasmissione di messaggi corti, ma se la quantità di dati da trasmettere è elevata allora la rete non è più efficiente.

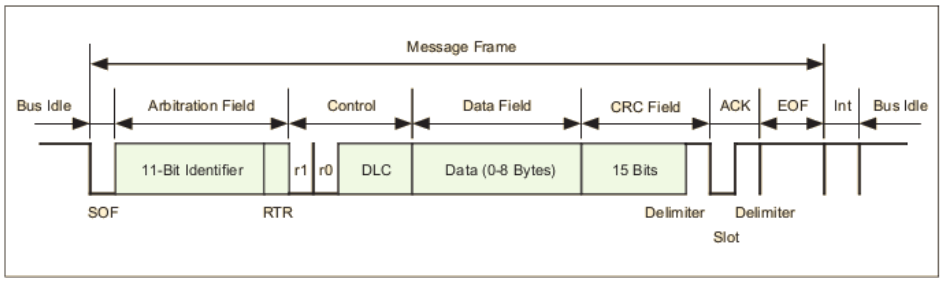

Figura 1.3: Struttura del pacchetto DeviceNet.

Capitolo 2

Parametri di rete

I parametri di rete descrivono quanto bene questa assolve alle sue funzioni. Tra questi c'è l'ampiezza di banda, cioè il numero di bit che può essere trasmesso per secondo e la velocità, che è l'inverso del bit rate (T_{bit}) che indica il tempo per trasmettere un bit attraverso la rete.

Altri parametri sono l'affidabilità e la sicurezza. Alcune reti sono fisicamente più vulnerabili rispetto ad altre alla corruzione di dati a causa delle interferenze elettromagnetiche e, per aumentarne l'affidabilità, vengono inviati segnali di acknowledgment. Se il segnale di acknowledgment non viene ricevuto viene rinviato. Questa tecnica però aggiunge overhead ai pacchetti e quindi diminuisce l'ampiezza di banda destinata alla trasmissione di dati. La sicurezza è un altro fattore che deve essere considerato, specialmente quando le reti sono vulnerabili ad attacchi via internet o virus. La maggior parte dei sistemi industriali non sono progettati per garantire un'alta sicurezza perché normalmente sono fisicamente isolati e non sono collegati via internet.

I parametri più studiati sono il tempo di trasmissione, tempo di ritardo e jitter, che indica come varia il ritardo. In seguito vengono descritte dettagliatamente queste ultime caratteristiche di rete, confrontando come variano in base al *Media Access Control* usato.

2.1 Tempo di ritardo

Il tempo di ritardo, T_{delay}, è definito come la differenza tra l'istante in cui il nodo sorgente comincia ad inviare un messaggio e l'istante in cui il nodo destinatario completa la ricezione di tale messaggio.

Il tempo totale di ritardo può essere diviso in tre parti: tempo di ritardo alla sorgente, nel canale di rete e al nodo destinatario, come mostrato in Figura 2.1. Il tempo di ritardo alla sorgente include il preprocessing time, T_{pre}, che è il tempo di computazione e dipende dal software, dalle caratteristiche di hardware del dispositivo, e dal tempo di attesa, T_{wait}. Il quale a sua volta è dato dal tempo di coda, T_{queue}, e dal blocking time, T_{block}. Il

tempo di attesa può diventare significativo ed è correlato alla quantità di dati che la sorgente deve inviare e dal traffico di rete. Il tempo di ritardo del canale, T_{tx}, comprende il tempo totale di trasmissione del messaggio lungo il canale, T_{frame}, e il ritardo di propagazione nella rete, T_{prop}. Questi dipendono dalla dimensione del messaggio, dal data rate e dalla lunghezza del cavo della rete. Infine il tempo di ritardo al nodo di destinazione è dato dal tempo di postprocessing, T_{post}, che include il tempo necessario al dispositivo per decodificare il segnale ricevuto. Il tempo di ritardo può essere esplicitato come segue:

$$\begin{aligned} T_{delay} &= T_{pre} + T_{wait} + T_{tx} + T_{post} \\ &= T_{pre} + T_{queue} + T_{block} + T_{frame} + T_{prop} + T_{post} \end{aligned} \tag{2.1}$$

Preprocessing e postprocessing time sono tipicamente costanti e dipendono da processi computazionali piuttosto che dalla rete e dai protocolli. Quando nella sorgente più messaggi attendono di essere inviati, questi vengono memorizzati in un buffer e il tempo di attesa in coda è T_{queue}. Esso dipende dal blocking time dei precedenti messaggi e dalla periodicità con cui questi messaggi vengono inviati.

Nelle seguenti sezioni vengono analizzati il blocking time, il frame time e il tempo di propagazione per ognuno dei tre control network analizzati precedentemente.

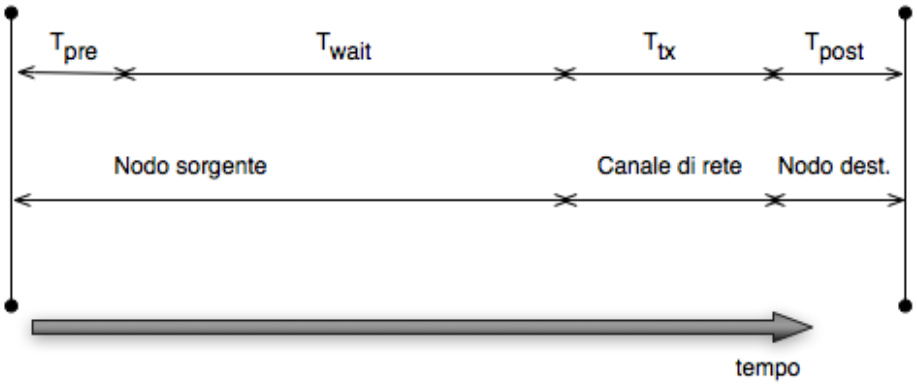

Figura 2.1: Diagramma temporale del tempo di ritardo.

2.2 Blocking Time

Il blocking time, che è il tempo che un messaggio deve attendere quando il nodo è pronto per spedirlo, dipende dal protocollo di rete ed è il maggior fattore di determinismo nella prestazione di rete. Include il tempo di coda

dovuto ad altri nodi che stanno usando la rete e il tempo necessario per rinviare il messaggio quando avvengono collisioni.

Ethernet

Il blocking time per l'Ethernet comprende il tempo di ritardo dovuto alle collisioni e il tempo atteso prima di riprovare ad inviare il messaggio. Dall'algoritmo BEB si può determinare il tempo di attesa in modo probabilistico, quindi l'esatta analisi del blocking time è estremamente difficile. Il valore atteso del blocking time può essere descritto come segue:

$$E\{T_{block}\} = \sum_{k=1}^{16} E\{T_k\} + T_{resid}$$

dove T_{resid} denota il tempo residuo affinchè la rete sia libera, mentre $E\{T_k\}$ è il valore atteso che avvenga la k-esima collisione. Dopo la sedicesima collisione il nodo interrompe i tentativi di trasmissione e lo segnala ai livelli superiori con un messaggio d'errore. È chiaro che T_{block} non può essere determinato con precisione, né si può calcolarne un limite superiore a causa dell'interruzione di trasmissione al sedicesimo tentativo.

ControlNet

Nel protocollo ControlNet, se un nodo vuole inviare un messaggio deve attendere di ricevere il token dal nodo che lo precede. Pertanto il blocking time, che dipende dal tempo di trasmissione dei precendenti nodi e dal tempo di rotazione del token, può essere espresso come segue:

$$T_{block} = T_{resid} + \sum_{j \in N_{nonqueue}} T_{token}^{(j)} + \sum_{j \in N_{queue}} \min\left(T_{tx}^{(j,n_j)}, T_{node}\right) + T_{guard}$$

dove T_{resid} è il tempo residuo necessario al nodo corrente per finire la trasmissione, $N_{nonqueue}$ e N_{queue} denotano l'insieme di nodi, rispettivamente, con e senza messaggi in coda, e T_{guard} è l'intervallo di tempo in cui il canale rimane libero per garantire che non avvengano interferenze tra le varie trasmissioni. Il numero di messaggi in coda al j-esimo nodo è indicato con n_j, mentre T_{node} è il tempo massimo (i.e., token holding time) assegnato a ciascun nodo che utilizza interamente il canale. T_{token} è il token-passing time, che dipende dal tempo necessario per trasmettere il token e la sua propagazione dal nodo $i-1$ al nodo i. Nel ControlNet T_{token} è dato semplicemente dalla somma di T_{frame} con una lunghezza di dati inviati pari a zero e da T_{prop}. Se un nuovo messaggio è in coda ad un nodo mentre questo nodo ha il token, allora $T_{block} = T_{tx}^{(j,n_j)}$, dove j è il numero del nodo.

ControlNet è una rete deterministica perché si può calcolare il tempo massimo di ritardo. Se si conosce il periodo di ogni nodo e i messaggi che

devono essere inviati, quindi si può esplicitare n_j e gli insiemi $N_{nonqueue}$ e N_{queue}, allora T_{block} può essere calcolato con precisione.

DeviceNet

Se il blocking time, T_{block}, in una DeviceNet è finito, allora può essere calcolato iterando la seguente equazione da $k = 1$ fino alla sua convergenza:

$$T_{block}^{(k)} = T_{resid} + \sum_{j \in N_{hp}} \left\lceil \frac{T_{block}^{(k-1)} + T_{bit}}{T_{peri}^{(j)}} \right\rceil T_{tx}^{(j)}$$

dove T_{resid} è il tempo residuo necessario al nodo corrente per finire la trasmissione, N_{hp} è l'insieme di nodi con priorità superiore rispetto al nodo in attesa, $T_{peri}^{(j)}$ è il periodo del j-esimo nodo, e $\lceil x \rceil$ denota il più piccolo intero maggiore o uguale a x. La sommatoria denota il tempo necessario a inviare i messaggi con priorità maggiore.

Per un nodo con bassa priorità, mentre aspetta che il canale diventi disponibile, è possibile che altri nodi con priorità più alta entrino in coda, in questo caso il nodo con bassa priorità perde nuovamente la possibilità di trasmettere. A causa di questo meccanismo di priorità, il blocking time di un messaggio con bassa priorità non può essere determinato deterministicamente, né calcolato un limite superiore.

2.3 Frame Time

Il frame time, T_{frame}, dipende dalla dimensione dei dati dell'overhead e di possibili padding e dal tempo di bit. Sia N_{data} la dimensione dei dati in byte, N_{ovhd} il numero di byte usati per l'overhead, N_{pad} il numero di byte usati per ottenere le dimensioni minime di pacchetto, e N_{stuff} il numero di byte usati per meccanismi di stuffing usati in alcuni protocolli (i.e., in DeviceNet se più di cinque bit consecutivi hanno il valore 1, allora uno 0 viene aggiunto e viceversa). Il frame time può quindi essere espresso come:

$$T_{frame} = [N_{data} + N_{ovhd} + N_{pad} + N_{stuff}] \times 8 \times T_{bit} \tag{2.2}$$

I valori N_{data}, N_{ovhd}, N_{pad} e N_{stuff} possono essere determinati in modo esplicito dalla struttura di pacchetto.

2.4 Tempo di propagazione

Il tempo di propagazione, T_{prop}, dipende dalla velocità di trasmissione e dalla distanza tra il nodo sorgente e il destinatario. Nel peggior caso, il ritardo di propagazione tra un nodo ed un altro nelle tre control network sono $T_{prop} = 25.6\mu s$ per l'Ethernet (2500 m), $T_{prop} = 10\mu s$ per il ControlNet

(1000 m) e $T_{prop} = 1\mu s$ per il DeviceNet (100 m). La distanza tra parentesi indica la lunghezza massima di cavo usata. Il tempo di propagazione non è facilmente calcolabile perché la distanza tra nodo sorgente e destinatario varia a seconda della trasmissione in atto.

Capitolo 3

Controllore

Nei precedenti capitoli sono stati studiati i problemi che insorgono nelle comunicazioni attraverso la rete, però, dato che vengono utilizzati i pacchetti per la trasmissione dei dati, è importante considerare anche l'insorgere di problemi quali la perdita di pacchetti o l'arrivo dei pacchetti in ordine errato.

Ci sono due metodi per affrontare tutti questi problemi nella progettazione di un controllore adeguato per la NCS. Il primo è realizzare un controllore senza considerare il ritardo o la perdita dei pacchetti, ma che sia in grado di minimizzare la probabilità che questi eventi avvengano. Per esempio, sono stati proposti diversi algoritmi sul controllo della congestione, in grado di migliorare le prestazioni di una rete quando il traffico è superiore al limite che la rete stessa può gestire. L'altro approccio è di trattare il protocollo di rete e il traffico come condizioni prestabilite. Considerando le caratteristiche della rete nella progettazione del controllore si è in grado di ottenere prestazioni migliori. Inoltre, quando avviene una collisione di pacchetto, invece di cercare di ristabilire il pacchetto mancante, può essere vantaggioso abbandonare il pacchetto e mandarne uno nuovo.

Nel prossimo paragrafo viene introdotto lo studio di un sistema retroazionato con quest'ultimo approccio. In seguito, viene introdotta una tecnica alternativa, chiamata *packet-based*, che consente al controllore di spedire più valori utili di controllo in un singolo pacchetto in modo da sfruttare al meglio la rete e permettere all'attuatore di determinare il segnale di controllo più adeguato da applicare al sistema.

3.1 Progetto del controllore

La rete può essere modellata come un sistema lineare a tempo discreto:

$$\begin{cases} x(k+1) = Ax(k) + Bu(k) \\ y(k) = Cx(k) \end{cases} \tag{3.1}$$

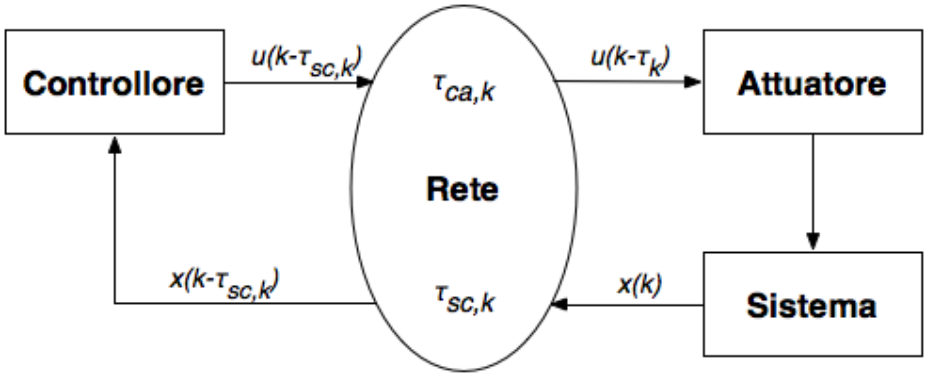

Figura 3.1: Schema a blocchi di un Networked Control System.

dove $x(k) \in \mathbb{R}^n$, $u(k) \in \mathbb{R}^m$, $y(k) \in \mathbb{R}^r$, $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times m}$ e $C \in \mathbb{R}^{r\times n}$.

La legge di retroazione è comunemente ottenuta come segue:

$$u(k) = Kx(k) \tag{3.2}$$

dove il guadagno K è tempo invariante.

Però la presenza di ritardi nella rete, perdita di pacchetti o l'arrivo dei pacchetti in disordine non rendono adeguato il controllore definito in (3.2). L'influenza di questi vincoli nella progettazione del controllore è analizzata come segue.

- *Ritardo introdotto dalla rete:* nel controllore bisogna tenere conto del ritardo introdotto dalla rete che può essere diviso nel ritardo sistema-contollore ($\tau_{sc,k}$) e ritardo controllore-attuatore ($\tau_{ca,k}$), come mostrato in Figura 3.1. Sia $\tau_k = \tau_{sc,k} + \tau_{ca,k}$ l'intero ritardo al tempo k. La legge di retroazione, per tenere conto dei ritardi, può essere modificata come segue:
$$u(k) = Kx(k - \tau_k) \tag{3.3}$$
si noti che il guadagno K è ancora tempo invariante.

- *Perdita di pacchetti:* ci sono due semplici strategie per affrontare il problema. La prima, chiamata hold-input, consiste nell'usare il valore di controllo precedente nel caso il pacchetto non arrivi:

$$u(k) = \begin{cases} \bar{u}(k), & \text{se la trasmissione avviene correttamente,} \\ u(k-1), & \text{atrimenti.} \end{cases}$$

Oppure ponendo il controllore a zero in caso di errore, con la strategia zero-input:

$$u(k) = \begin{cases} \bar{u}(k), & \text{se la trasmissione avviene correttamente,} \\ 0, & \text{atrimenti.} \end{cases}$$

dove $\bar{u}(k)$ è il segnale di controllo all'istante k.

- *Disordine nell'arrivo dei pacchetti:* nelle NCSs, differenti pacchetti sono trasmessi con differenti ritardi che possono produrre situazioni tali che, pacchetti inviati prima, arrivino a destinazione dopo rispetto a pacchetti successivi. Il modo più semplice per affrontare il problema è scartare i pacchetti che arrivano in disordine, aumentando così il numero di pacchetti persi. Per questa ragione, in seguito i pacchetti disordinati non saranno trattati in modo particolare, ma saranno considerati parte dei pacchetti persi.

Tutti questi effetti negativi nella comunicazione rendono il controllore definito in (3.2) non appropiato per il progetto di una NCS. La scelta migliore si basa nel seguente contollore:

$$u(k) = K_{\tau_{sc,k},\tau_{ca,k}} x(k - \tau_k) \tag{3.4}$$

dove, per differenti condizioni di rete, vengono applicati differenti guadagni.

3.2 Controllo Packet-Based

In una Networked Control System trasmettere un segnale di controllo o trasmetterne molti consuma le stesse risorse di rete. A partire da questa osservazione si è progettato il controllore packet-based che, senza ulteriori requisiti, è in grado di soddisfare le esigenze della rete. Come verrà presentato in seguito, questo controllore è in grado di affrontare simultaneamente i tre problemi descritti precedentemente: ritardo, perdita e disordine dei pacchetti.

Prima di tutto si presentano le seguenti ipotesi necessarie per implementare il controllo packet-based:

1. Tutte le componenti del sistema, inclusi sensori, controllori e attuatori sono sincronizzate.

2. Tutti i pacchetti mandati sia dai sensori che dai controllori sono marcati temporalmente.

3. La somma del massimo ritardo introdotto dalla rete τ_{ca} (e rispettivamente τ_{sc}) e il massimo numero di pacchetti consecutivi persi, includendo i pacchetti eliminati perché in disordine, sono limitati superiormente da $\bar{\tau}_{ca}$ ($\bar{\tau}_{sc}$) e

$$\bar{\tau}_{ca} \leq \frac{B_{packet}}{B_{control}} - 1 \tag{3.5}$$

dove B_{packet} è la dimensione effettiva dei dati nel pacchetto e $B_{control}$ sono i bit richiesti per codificare un singolo segnale di controllo.

Quest'ultima ipotesi per i protocolli standard è facile da soddisfare. Mentre per quanto riguarda le ipotesi 1 e 2, esse sono richieste affinché, sia il controllore che l'attuatore, siano in grado di conoscere il ritardo dei pacchetti introdotto dalla rete, appena questi arrivano.

Grazie alle assunzioni fatte, sono proposti i segueti schemi per compensare il ritardo introdotto dalla rete e la perdita di pacchetti (come pure il disordine).

Compensazione del ritardo

Per compensare il ritardo di entrambi i canali sfruttando i vantaggi della trasmissione packet-based, si progetta, oltre al controllore packet-based, un *Control Action Selector* nell'attuatore, in grado di scegliere il segnale di controllo più appropiato da applicare al sistema.

- *Controllore Packet-Based*: il segnale ricevuto dal controllore al tempo k è definito da $x(k - \tau_{sc,k})$, dove $\tau_{sc,k}$ indica il ritardo del pacchetto introdotto dalla rete nel canale sistema-controllore (vedi Fig. 3.1). A partire da questo stato, si ottiene il seguente controllore, come in (3.4):

$$u(k+i|k-\tau_{sc,k}) = K_{\tau_{sc,k},i}x(k-\tau_{sc,k}), \qquad i = 0, 1, 2, ..., \bar{\tau}_{ca} \quad (3.6)$$

  che può essere scritta in forma matriciale come:

$$U(k|k-\tau_{sc,k}) = \begin{bmatrix} u(k|k-\tau_{sc,k}) \\ u(k+1|k-\tau_{sc,k}) \\ \vdots \\ u(k+\bar{\tau}_{ca}|k-\tau_{sc,k}) \end{bmatrix} \quad (3.7)$$

  Si nota che ad ogni istante temporale non viene elaborato un singolo segnale di controllo, da qui il nome controllore packet-based.

  Dall'ipotesi 3, la sequenza di controllo $U(k|k - \tau_{sc,k})$ può essere raggruppata in un unico pacchetto e inviata all'attuatore. Come detto in precedenza, inviare una sequenza di segnali di controllo invece che uno solo non aumenta il traffico della rete, così si riesce a migliorare le prestazioni di rete, come si dimostra in seguito.

- *Control Action Selector:* per annullare gli effetti del ritardo, si progetta nell'attuatore il cosiddetto Control Action Selector, in grado di memorizzare una sequenza di controllo alla volta.

Ad ogni istante di esecuzione, l'attuatore sceglie l'appropiato segnale di controllo in grado di compensare il ritardo del canale controllore-attuatore, grazie al Control Action Selector, per poi applicarlo al sistema retroazionato. In questo modo si riesce ad annullare gli effetti del ritardo in entrambi i canali.

Si noti che i ritardi introdotti dai canali sono rispettivamente $\tau_{ca,k}$ e $\tau_{sc,k}$ e la sequenza di controllo nel canale controllore-attuatore usata dall'attuatore al tempo k può essere rappresentata come:

$$U(k-\tau_{ca,k}|k-\tau_k) = \begin{bmatrix} u(k-\tau_{ca,k}|k-\tau_k) \\ u(k-\tau_{ca,k}+1|k-\tau_k) \\ \vdots \\ u(k|k-\tau_k) \\ \vdots \\ u(k+\bar{\tau}_{ca}|k-\tau_k) \end{bmatrix} \tag{3.8}$$

e il segnale che effettivamente viene applicato al sistema è proprio $u(k|k-\tau_k)$. È importante sottolineare che, grazie all'ipotesi 3, il segnale di controllo appropriato è sempre disponibile.

Compensazione nella perdita (o disordine) dei pacchetti

Per affrontare il problema della perdita di pacchetti, o del loro disordine, si effettua un processo di confronto nel Control Action Selector. Quando arriva un pacchetto al Control Action Selector, questo non viene immediatamente sostituito con il pacchetto presente in precedenza, a causa della possibile perdita (o disordine) di pacchetti.

Indicando con $U(k_1-\tau_{ca,k_1}|k_1-\tau_{k_1})$ la sequenza di controllo presente del Control Action Selector e con $U(k_2-\tau_{ca,k_2}|k_2-\tau_{k_2})$ quella in arrivo, il processo di confronto è determinato dalla seguente regola:

$$U(k-\tau^*_{ca,k}|k-\tau^*_k) = \begin{cases} U(k_2-\tau_{ca,k_2}|k_2-\tau_{k_2}), & \text{se } k_1-\tau_{k_1} < k_2-\tau_{k_2}, \\ U(k_1-\tau_{ca,k_1}|k_1-\tau_{k_1}), & \text{atrimenti.} \end{cases} \tag{3.9}$$

dove l'apice * è usato per indicare il ritardo introdotto dalla rete dell'ultimo segnale di controllo contenuto nel Control Action Selector, dopo che è avvenuto il confronto. Come conseguenza del processo di confronto, la sequenza di controllo contenuta nel Control Action Selector è sempre l'ultima disponibile per ogni istante temporale.

Si può riassumere il metodo packet-based per le Networked Control Systems con il seguente algoritmo:

P1. All'istante k, se il controllore packet-based non riceve lo stato $x(k - \tau_{sc,k})$, allora $k = k + 1$, altrimenti fai i passi P1a-P1c;

P1a. Leggi il ritardo introdotto dal canale sistema-controllore, $\tau_{sc,k}$;

P1b. Calcola la sequenza di controllo $U(k|k - \tau_{sc,k})$ usando (3.7) e (3.6);

P1c. Spedisci all'attuatore, in un pacchetto, $U(k|k - \tau_{sc,k})$ e i valori k e $\tau_{sc,k}$;

P2. Aggiorna il Control Action Selector usando (3.9) quando arrivano i pacchetti all'attuatore;

P3. Applica $u(k|k - \tau_k^*)$ al sistema.

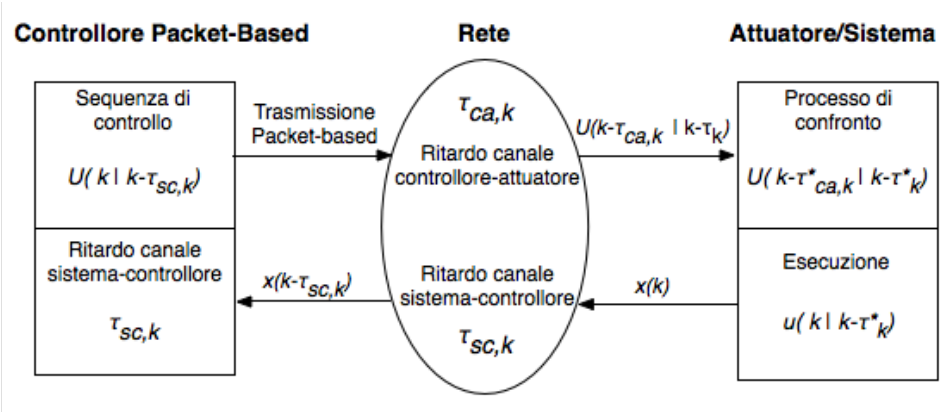

Figura 3.2: Struttura schematica del processo *packet-based*.

Capitolo 4

Simulazione

In questo capitolo viene simulata una Networked Control System nel controllo del pitch di un aeromobile. Il pitch, o beccheggio, è l'oscillazione di un veivolo attorno al proprio asse trasversale.

Vengono dapprima studiate le equazioni generali che governano il moto di un aeromobile, poi si costruisce un modello a tempo discreto con un controllore che stabilizza il sistema. Infine viene analizzato come varia la stabilità qualora avvenga una perdita di pacchetti.

4.1 Modello del sistema

Le equazioni che governano il movimento di un aeromobile sono molto complicate, sono costituite da sei equazioni differenziali non lineari. Ciò nonostante, sotto certe condizioni, le equazioni possono essere linearizzate e divise in equazioni longitudinali e laterali. Il pitch di un aeromobile è determinato dalle equazioni longitudinali. Si ipotizzi che l'aereo sia ad altitudine e velocità costante, in questo caso le forze negli assi x e y sono bilanciate (vedi Fig. 4.1). Inoltre si semplifichi ulteriormente il problema assumendo che quando varia l'angolo di pitch, la velocità non presenti variazioni. Sotto queste ipotesi le equazioni che governano il moto dell'aeromobile possono essere rappresentate come segue:

$$\begin{cases} \dot{\alpha} = \mu\Omega\sigma[-(C_L + C_D)\alpha + \frac{1}{\mu - C_L}q - (C_W \sin\gamma)\theta + C_L] \\ \dot{q} = \frac{\mu\Omega}{2i_{yy}}[[C_M - \eta(C_L + C_D)]\alpha + [C_M + \sigma C_M(1 - \mu C_L)]q + (\eta C_W \sin\gamma)\delta] \\ \dot{\theta} = \Omega q \end{cases} \tag{4.1}$$

dove α =angolo di attacco, q =indice di pitch, θ =angolo di pitch, δ =angolo di deviazione, $\mu = \frac{\rho S\bar{c}}{4m}$, ρ =densità dell'aria, S =area dell'ala, $\bar{c}$ =lunghezza media della corda alare, m =massa dell'aeromobile, $\Omega = \frac{2U}{\bar{c}}$, U =velocità di volo, C_T =coefficiente di spinta, C_D =coefficiente di traino, C_L =coefficiente

di sollevamento, C_W =coefficiente di peso, C_M =coefficiente di pitch istantaneo, γ =angolo di traiettoria di volo, $\sigma = \frac{1}{1+\mu C_L}$, i_{yy} =momento d'inerzia normalizzato e $\eta = \mu\sigma C_M$.

Per lo studio del pitch, l'input è rappresentato dall'angolo di deviazione δ, mentre l'output è l'angolo di pitch θ.

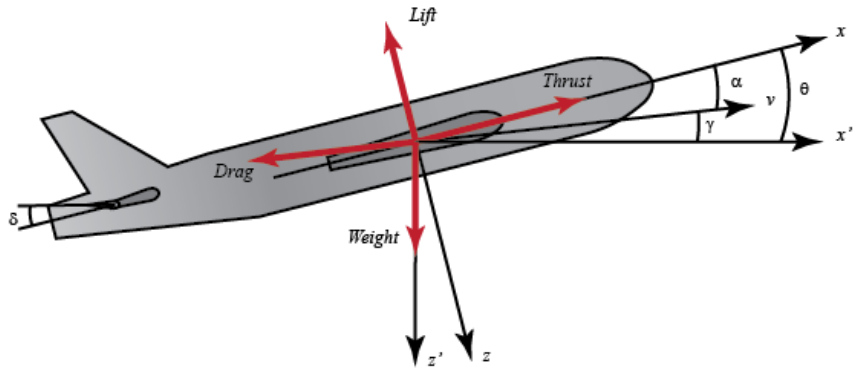

Figura 4.1: Coordinate degli assi e forze in gioco in un aeromobile.

Per semplificare il modello si possono sostituire alcuni dati con i valori numerici prendendo come riferimento un Boeing commerciale, ottenendo:

$$\begin{cases} \dot{\alpha} = -0.313\alpha + 56.7q + 0.232\delta \\ \dot{q} = -0.0139\alpha - 0.426q + 0.0203\delta \\ \dot{\theta} = 56.7q \end{cases} \tag{4.2}$$

Il precedente modello si può riscrivere in forma matriciale:

$$\begin{bmatrix} \dot{\alpha} \\ \dot{q} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} -0.313 & 56.7 & 0 \\ -0.0139 & -0.426 & 0 \\ 0 & 56.7 & 0 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix} + \begin{bmatrix} 0.232 \\ 0.0203 \\ 0 \end{bmatrix} \begin{bmatrix} \delta \end{bmatrix}$$

$$y = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha \\ q \\ \theta \end{bmatrix}$$

4.2 Modello a tempo discreto

Il modello analizzato è a tempo continuo però il controllore di una NCS è a tempo discreto. Bisogna quindi trasformare il segnale determinato dalle

equazioni (4.2) in un sistema a tempo discreto, e per fare ciò è necessario campionare il segnale e trasformarlo da analogico a digitale, come mostrato in Figura 4.2.

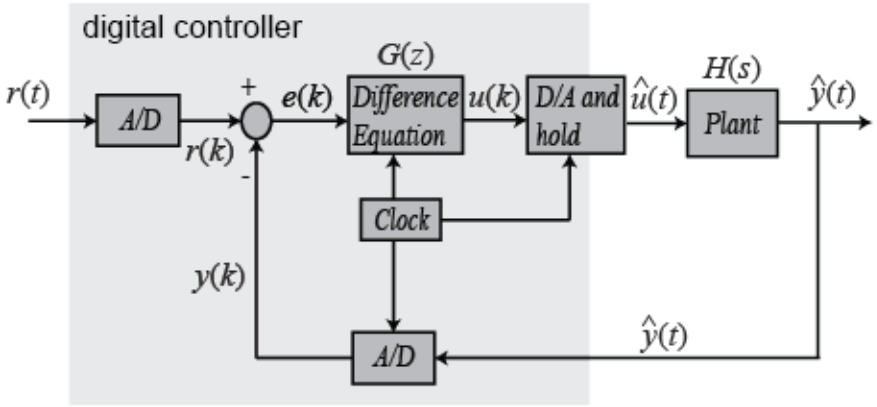

Figura 4.2: Schema a blocchi di un controllore digitale.

Per sincronizzare i convertitori A\D e D\A è essenziale usare un clock con una frequenza di campionamento T. Una buona regola per scegliere la frequenza di campionamento è prendere almeno 1/30esimo della banda passante della funzione di trasferimento del sistema retroazionato.

Applicando la trasformata di Laplace alle equazioni (4.2) si ottiene:

$$\begin{cases} sA(s) = -0.313A(s) + 56.7Q(s) + 0.232\Delta(s) \\ sQ(s) = -0.0139A(s) - 0.426Q(s) + 0.0203\Delta(s) \\ s\Theta(s) = 56.7Q(s) \end{cases} \tag{4.3}$$

E la funzione di trasferimento è dunque:

$$F(s) = \frac{\Theta(s)}{\Delta(s)} = \frac{1.151s + 0.1774}{s^3 + 0.739s^2 + 0.921s} \tag{4.4}$$

Per ottenere la funzione di trasferimento del sistema retroazionato si può usare la funzione MATLAB:

G_pitch = feedback(F_pitch,1);

dove F_pitch è la funzione di trasferimento, mentre G_pitch è quella retroazionata. Si ottiene quindi:

$$G(s) = \frac{F(s)}{1 + F(s)} = \frac{1.151s + 0.1774}{s^3 + 0.739s^2 + 2.072s + 0.1774} \tag{4.5}$$

Ora si possono ricavare i diagrammi di Bode con il comando:

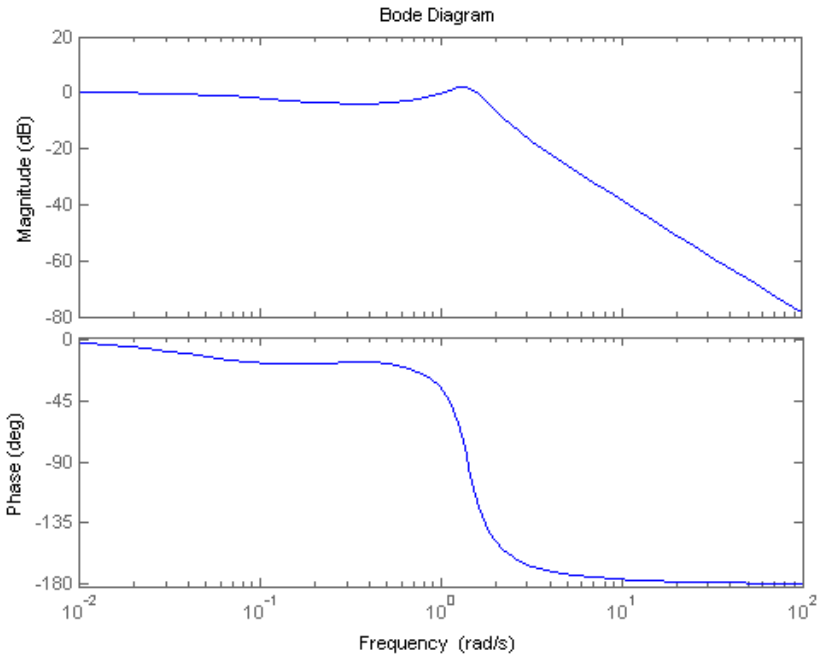

Figura 4.3: Diagramma di Bode.

bode(G_pitch);
e si ottengono i grafici in Fig. 4.3.

Dai grafici di Bode, si vede che la banda passante può essere approssimata a 2 rad/s. Allora, per avere un tempo di campionamento sufficientemente piccolo, si può scegliere 1/100 s.

A questo punto si può applicare la funzione MATLAB c2d:

sys_d = c2d(sys_ss,Ts,'zoh');
che in uscita dà un modello a tempo discreto, dati in ingresso un modello continuo, il tempo di campionamento e il metodo di conversione usato. In questo caso è stato usato il mantenitore di ordine zero (*Zero-order hold,* ZOH), che ricostruisce il segnale, da digitale ad analogico, mantenendolo costante negli intervalli di tempo in cui non viene campionato.

Nel caso dell'aereomobile il sistema a tempo discreto che si ottiene è:

$$\begin{bmatrix}\alpha(k+1)\\ q(k+1)\\ \theta(k+1)\end{bmatrix} = \begin{bmatrix}0.9969 & 0.05649 & 0\\ -0.0001 & 0.9957 & 0\\ 0 & 0.5658 & 1\end{bmatrix}\begin{bmatrix}\alpha(k)\\ q(k)\\ \theta(k)\end{bmatrix} + \begin{bmatrix}0.0024\\ 0.0002\\ 0.0001\end{bmatrix}\begin{bmatrix}\delta(k)\end{bmatrix}$$

$$y(k) = \begin{bmatrix}0 & 0 & 1\end{bmatrix}\begin{bmatrix}\alpha(k)\\ q(k)\\ \theta(k)\end{bmatrix} + \begin{bmatrix}0\end{bmatrix}\begin{bmatrix}\delta(k)\end{bmatrix}$$

Indicando con $x(k)$ la matrice delle variabili di stato $[\alpha, q, \theta]$, e con A, B, C, D le matrici numeriche, il sistema si riscrive nel modo seguente:

$$\begin{cases} x(k+1) = Ax(k) + B\delta(k) \\ \theta(k) = Cx(k) \end{cases} \tag{4.6}$$

Ora, come mostrato in Fig. 4.4 si sostituisce $\delta(k)$ con $\delta(k) = \theta_{des}(k) - Kx(k)$, dove K è il guadagno del controllore e θ_{des} è il valore di pitch che si desidera avere. Il sistema quindi diventa:

$$\begin{cases} x(k+1) = (A - BK)x(k) + B\theta_{des}(k) \\ \theta(k) = Cx(k) \end{cases} \tag{4.7}$$

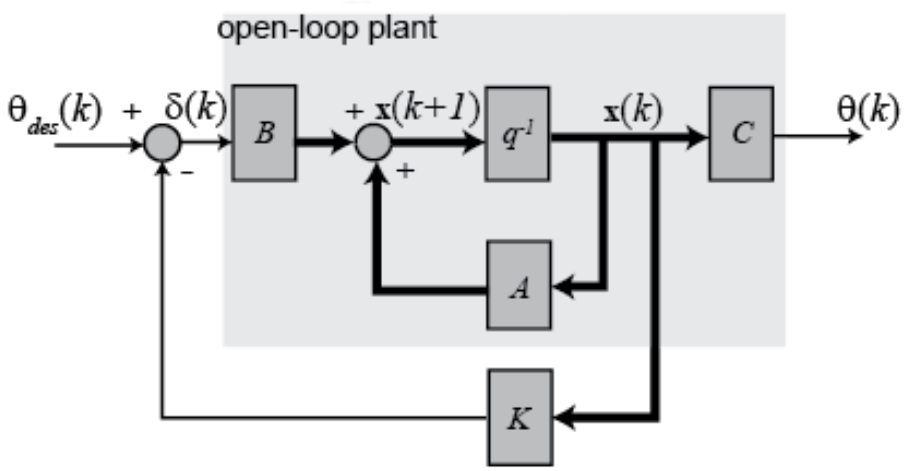

Figura 4.4: Schema del sistema retroazionato.

Per determinare la matrice di guadagno K si usa il metodo del regolatore lineare quadratico (LQR). Questo metodo permette di regolare $x(k)$ cercando di minimizzare la funzione costo:

$$J(\delta) = \sum_{k=0}^{+\infty} (x(k)^T Q x(k) + \delta(k)^T R \delta(k))$$

Nel caso specifico dell'aeromobile si assuma che le condizioni inziali siano poste a zero. Inoltre, per usare il metodo LQR, è necessario definire due parametri: la matrice state-cost weighted Q e la matrice control weighted R. Per semplicità si pone la matrice R pari a 1, metre la matrice $Q = pC^T C$, dove p indica il fattore di weighting e viene posto uguale a 50.

Implementando in MATLAB il comando:

[K] = dlqr (A,B,Q,R)

dove dlqr è la versione a tempo discreto del comando lqr, si ottiene la seguente matrice K:

$$K = \begin{bmatrix} -0.6436 & 168.3611 & 6.9555 \end{bmatrix}$$

Ora che si hanno tutti i dati a disposizione si può definire il sistema retroazionato di (4.7) con la funzione MATLAB:

sys_cl = ss(A−B∗K, B, C, D, Ts);

Nelle Figure 4.5 e 4.6 sono mostrati gli andamenti temporali dell'angolo di pitch, rispettivamente, senza e con sistema di retroazione. Si nota immediatamente che il sistema con anello aperto ha un andamento instabile, mentre dopo che è stato applicato il controllore il sistema diventa stabile e l'angolo di pitch a regime si stabilizza a circa 0.028 rad.

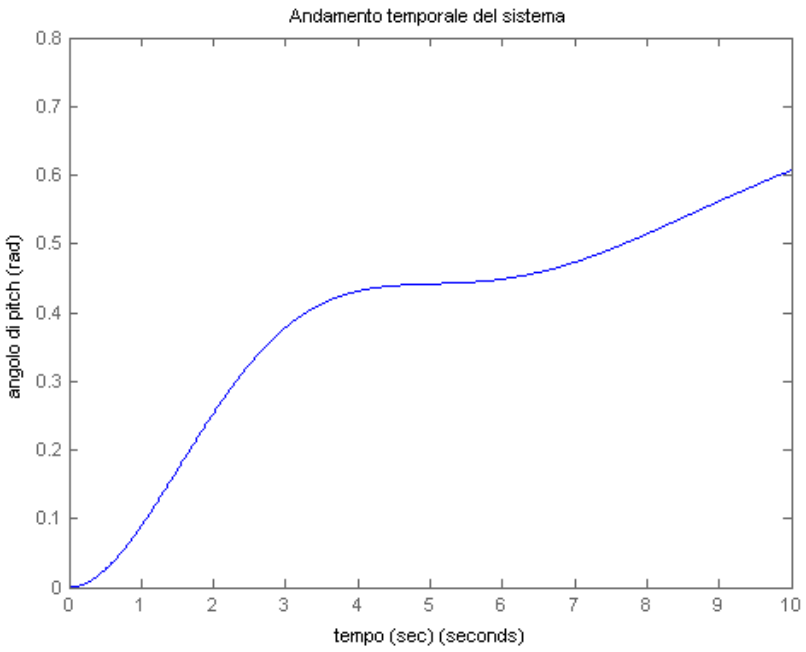

Figura 4.5: Evoluzione temporale dell'angolo di pitch nel sistema con anello aperto.

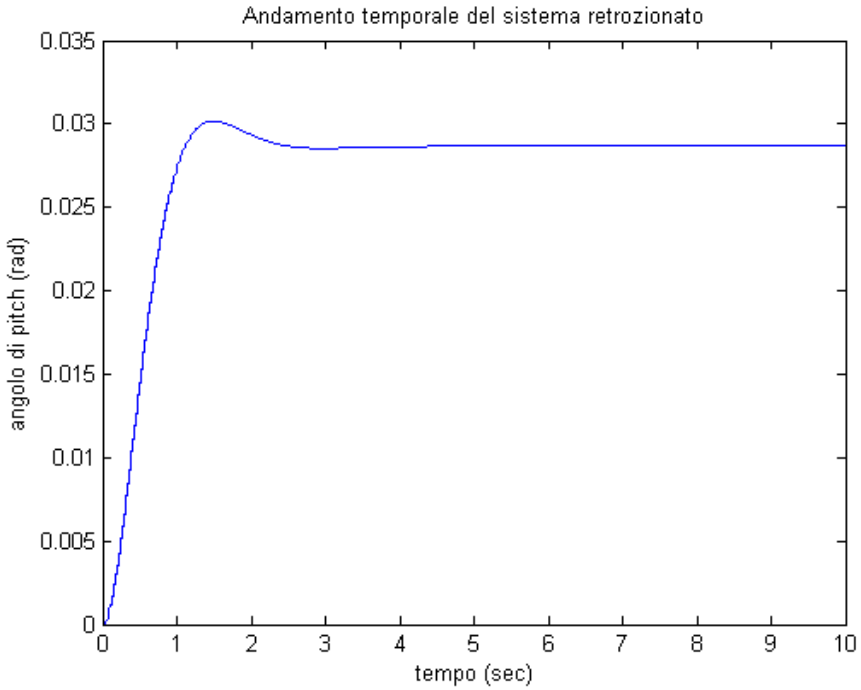

Figura 4.6: Evoluzione temporale dell'angolo di pitch nel sistema con anello chiuso.

4.3 Perdita di pacchetti

Come è stato analizzato nel Capitolo 3, quando si progetta il controllore di una Networked Control System bisogna considerare che le informazioni, raggruppate in pacchetti, viaggiano attraverso una rete e sono quindi soggette a problemi quali perdita di pacchetti e disordine nell'arrivo dei pacchetti. Questi due aspetti vengono affrontati nello stesso modo, infatti se un pacchetto arriva dopo rispetto ad un pacchetto inviato successivamente viene semplicemente scartato.

In seguito viene simulata nella NCS progettata per il pitch dell'aeromobile la perdita di pacchetti e analizzato il comportamento del sistema con le due strategie di controllo descritte nella Sezione 3.1: *zero-input* e *hold-input*.

Partendo dal sistema a tempo discreto (4.7), per semplicità si riscrive il modello per sottolineare il valore di controllo u:

$$\begin{cases} x(k+1) = Ax(k) + B\theta_{des}(k) - Bu(k) \\ u(k) = Kx(k) \end{cases} \tag{4.8}$$

A questo punto, il controllore viene modificato per considerare la probabilità di perdita di pacchetto. Viene quindi introdotta una nuova variabile che sarà

indicata con ν, essa vale 1 nel caso il pacchetto sia trasmesso correttamente, mentre vale 0 nel caso il pacchetto venga perso.

Nel caso della strategia zero-hold, se avviene una perdita di pacchetto tra il controllore e l'attuatore, il valore di controllo viene posto a zero $u^a(k) = 0$, mentre se la trasmissione avviene correttamente i valori di controllo dell'attuatore e del controllore sono uguali $u^a(k) = u^c(k)$ e assumono l'usuale valore determinato dal guadagno K. Il sistema diviene dunque:

$$\begin{cases} x(k+1) = Ax(k) + B\theta_{des}(k) - Bu^a(k) \\ u^a(k) = \nu(k)u^c(k) \\ u^c(k) = Kx(k) \end{cases} \tag{4.9}$$

Mentre se si utilizza la strategia hold-input, nel caso avvenga una perdita di pacchetti, quindi $\nu = 0$, il valore di controllo assume il valore precedente $u^a(k) = u^a(k-1)$, altrimenti se la trasmissione avviene corretamente $\nu = 1$ e, come prima, $u^a(k) = u^c(k)$. Il sistema può essere riscritto come:

$$\begin{cases} x(k+1) = Ax(k) + B\theta_{des}(k) - Bu^a(k) \\ u^a(k) = \nu(k)u^c(k) + (1-\nu(k))u^a(k-1) \\ u^c(k) = Kx(k) \end{cases} \tag{4.10}$$

Nella Fig. 4.7 è indicato il segnale con sistema retroazionato e i valori di controllo nel caso non ci siano perdite di pacchetti. In questo caso ovviamente le due strategie hanno identici valori di controllo dato che non ci sono dati mancanti.

Mentre le Figure 4.8, 4.9, 4.10 mostrano i risultati di tre simulazioni con diverse probabilità di perdita di pacchetto. Vengono confrontati gli andamenti dell'angolo di pitch assunti al variare del valore di $\nu(k)$, per le due strategie di controllo analizzate.

Si nota che l'angolo di pitch ha un andamento stabile anche con elevate perdite di pacchetto. I valori di controllo per le due strategie risultano molto differenti, infatti per lo hold-input i valori sono più regolari: il valore rimane costante per il periodo di tempo in cui $\nu = 0$ e quindi c'è stata perdita di pacchetti. Mentre in questi casi per lo zero-input il valore di controllo va a zero e risulta quindi avere andamenti più discontinui.

Nonostante ciò, i risultati sull'angolo di pitch sono simili per le due strategie, gli andamenti discostano solo quando le perdite di pacchetto sono molte. Inoltre si può notare che, con l'aumentare della percentuale di pacchetti persi, aumenta la sovraelongazione del segnale e aumenta l'angolo di pitch con cui si stabilizza il segnale a regime.

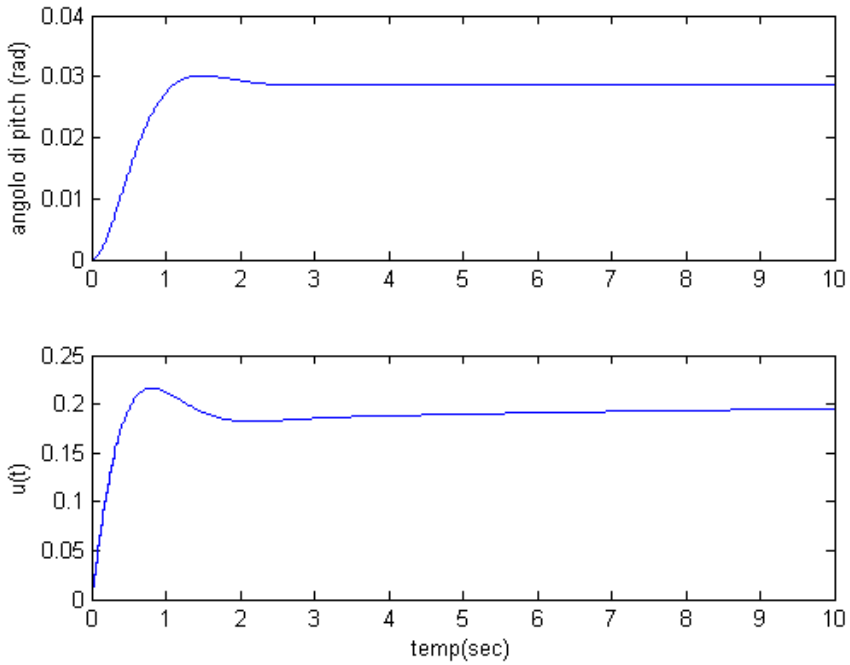

Figura 4.7: Segnale e valori di controllo senza perdita di pacchetto.

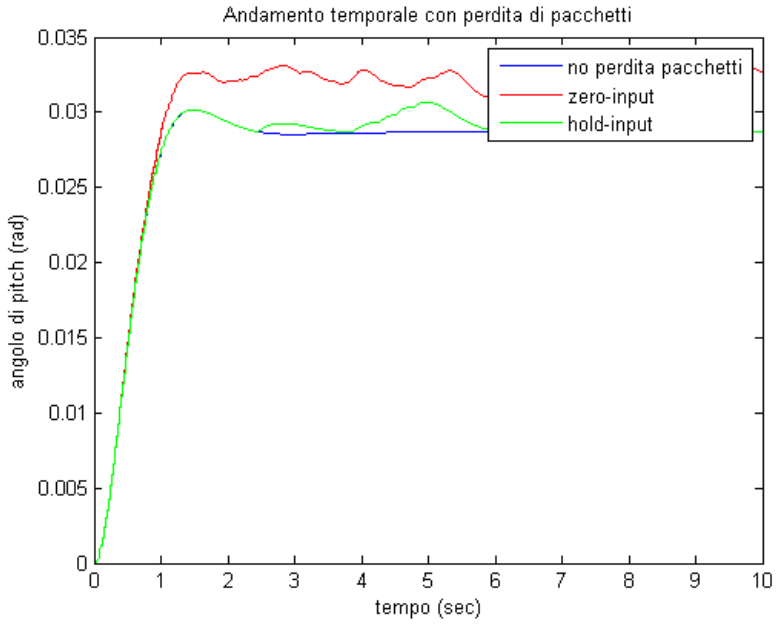

Figura 4.8: Probabilità di perdità di pacchetto del 10%.

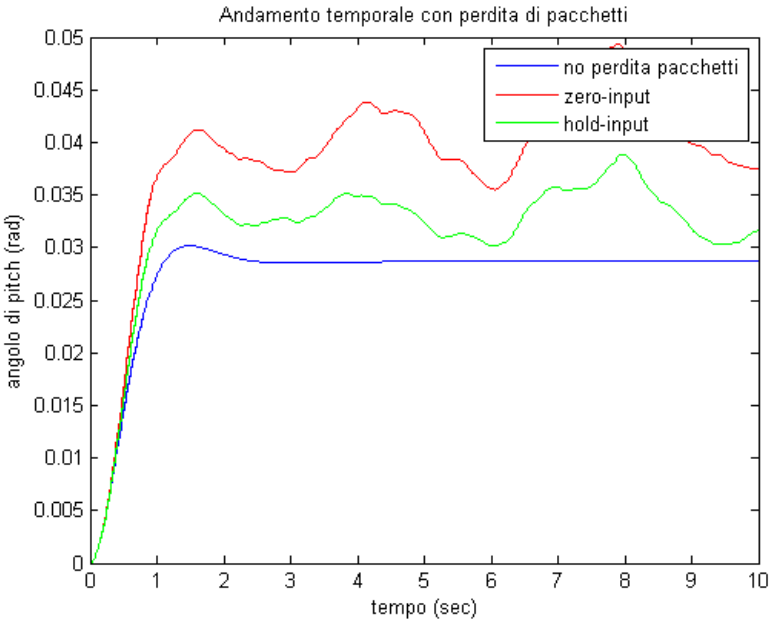

Figura 4.9: Probabilità di perdità di pacchetto del 30%.

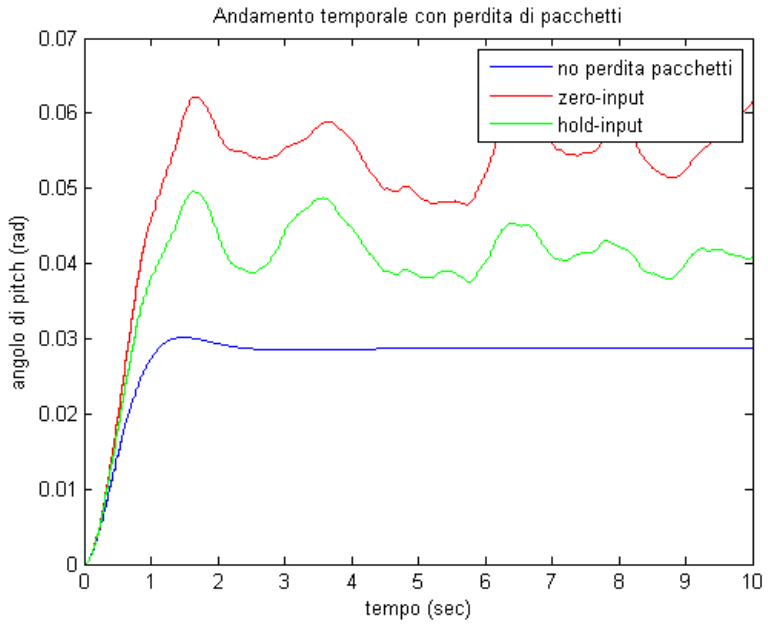

Figura 4.10: Probabilità di perdità di pacchetto del 50%.

Conclusioni

In questo lavoro di tesi si è cercato di descrivere gli aspetti principali e le potenzialità delle Networked Control Systems. Esse ormai sono assai diffuse in campo industriale, ma sono anche alla base di attuali studi su reti avanzate, come le smart grids o smart house.

Nello studio delle NCSs, dato che le informazioni viaggiano attraverso una rete, bisogna considerare aspetti quali rumori e congestione del traffico che compromettono la corretta trasmissione dei pacchetti. Nei primi Capitoli sono stati descritti i principali protocolli utilizzati in ambito industriale e ne sono stati analizzati i parametri di rete più significativi. Si è visto come le varie reti studiate abbiano valori molto differenti per parametri importanti come tempo di ritardo, tempo di propagazione o determinismo. Per questo è importante la scelta del più adeguato tipo di protocollo di rete che avviene considerando gli scopi e i requisiti che la rete deve avere.

Questi fattori devono essere considerati anche nella progettazione del controllore e dell'anello di retroazione. Inserire un controllore può migliorare notevolmente le prestazioni del sistema, si pensa a fattori quali stabilità, tempo di salita o assestamento. Ma è importante sfruttare l'utilizzo del controllore per migliorare anche gli aspetti negativi introdotti dalla rete. Nel Capitolo 3 è stato descritto come un controllore riesca a tener conto dei tre principali problemi dovuti alla rete: ritardo di rete, perdita di pacchetti e disordine nell'arrivo dei pacchetti. Con la descrizione del controllo Packet-Based è stato fatto un ulteriore passo; infatti questo tipo di controllore sfrutta la possibilità di poter inviare più valori di controllo nello stesso pacchetto, senza quindi consumare maggiori risorse di rete rispetto a quelle utilizzate nell'invio di un singolo valore di controllo. Il controllo Packet-Based è in grado di affrontare efficientemente i tre problemi detti precedentemente, migliorando così le prestazioni di rete.

I vantaggi che una Networked Control System può offrire si sono notati nel Capitolo 4 in cui è stato studiato dettagliatamente un modello che analizza il moto di un aeromobile per l'analisi dell'angolo di pitch. Prima di tutto, dato che il controllore è a tempo discreto, anche il modello è stato convertito da tempo contino a discreto. Con un valore di guadagno K adeguato si è notato che il modello instabile, diventava stabile. Ma per comprendere le potenzialità dell'uso di un controllore su una rete sono stati simulati gli

andamenti nel caso avvenga una perdita di dati. Questa perdita di dati rappresenta la mancata trasmissione di pacchetti o il forte ritardo che causa il disordine nell'arrivo dei pacchetti. Il problema è stato affrontato con due tipi di stategie messe a confronto: zero-input e hold-input. Dai grafici degli andamenti temporali dell'uscita del sistema si è potuto confermare che, anche con perdita di pacchetti, il sistema risultava stabile e gli andamenti continui. Le due stategie, nonostante agiscano in modi differenti e abbiano valori di controllo molto diversi erano in grado entrambi di dare risultati positivi. Nonostante nel caso analizzato il metodo hold-input aveva sovraelongazioni minori, e quindi era da preferirsi rispetto alla strategia zero-input, anche la scelta del controllore ottimale, come del protocollo di rete, non è univoca, ma dipende dal singolo sistema e dalle prestazioni che si vogliono ottenere.

Bibliografia

[1] N. Benvenuto, M. Zorzi, *Principles of Commuications Networks and Systems*, John Wiley & Sons, 2011.

[2] M. Bisiacco, M.E. Valcher, *Controlli Automatici: tutto quello che avreste voluto sapere a riguardo e non avete mai osato chiedere*, Edizioni Libreria Progetto, 2008.

[3] F. Lian, J. R. Moyne, D. M. Tilbury, *Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet*, IEEE Control Systems Magazine, Vol. 21, No. 1, pp. 66-83, 2001.

[4] W. Zhang, M. S. Branicky, S. M. Phillips, *Stability of Networked Control Systems*, IEEE Control Systems Magazine, Vol. 21, No. 1, pp. 84-99, 2001.

[5] J. R. Moyne, D. M. Tilbury, *The Emergence of Industrial Control Networks for Manufacturing Control, Diagnostic, and Safety Data*, Proceedings of the IEEE, Vol. 95, No. 1, pp. 29-47, 2007.

[6] Y. Zhao, G. Liu, D. Rees, *Design of a Packet-Based Control Framework for Networked Control Systems*, IEEE Transactions on Control Systems Technology, Vol. 17, No. 4, pp. 859-865, 2009.

[7] *http://ctms.engin.umich.edu/CTMS/index.php?aux=Home*, Control Tutorial for MATLAB and Simulink.