UNIVERSITA' DEGLI STUDI DI SIENA

FACOLTA' DI INGEGNERIA
CORSO DI LAUREA IN INGEGNERIA INFORMATICA

ANALISI E PROGETTO DI UN TELELABORATORIO PER IL CONTROLLO IN RETE DI PROCESSI DINAMICI

Relatore:

Chiar.mo Prof. Ing. Antonio Vicino

Correlatori:

Ing. Andrea Garulli

Ing. Domenico Prattichizzo

Tesi di laurea di:

Marco Casini

Anno Accademico 1998 - 99

Alla mia famiglia
per il costante incoraggiamento
che mi ha dato
in tutti questi anni di studio.

Indice

Introduzione

La diminuzione del costo dei calcolatori elettronici abbinata alla sempre maggiore capacità di calcolo, ha permesso un'ampia diffusione degli stessi nelle famiglie, nelle aziende e nei centri di ricerca. Inoltre, questi ultimi anni del secondo millennio, si distinguono in ambito informatico per l'avvento della rete Internet, una rete telematica in grado di collegare tra loro milioni di utenti, ed il cui sviluppo sta avendo una crescita ritenuta impensabile fino a qualche anno fa. Queste innovazioni hanno permesso di poter realizzare una nuova serie di servizi, quali ordinazioni e vendite di articoli via Internet, videoconferenze, operazioni chirurgiche a distanza, ecc.

Un altro aspetto di rilievo che queste tecnologie hanno messo a disposizione riguarda la possibilità da parte di soggetti dislocati in varie parti del mondo di poter comunicare tra di loro con estrema semplicità e costi ridotti. Questo aspetto assume ancor maggiore importanza se lo inquadriamo nell'ambito della ricerca scientifica [2]; fino a pochi anni fa, infatti, scienziati che lavoravano per un progetto comune erano costretti a comunicare tramite posta o telefono, oppure ad incontrarsi in un laboratorio centrale, il che comportava ingenti spese di viaggio e di stazionamento. Queste nuove tecnologie permettono oggi una comunicazione più veloce, più semplice e completamente indipendente dalla distanza. Tutto questo si traduce in un più rapido sviluppo ed in una più rapida divulgazione delle conoscenze di base, nonché in una riduzione del tempo che intercorre tra la scoperta di un nuovo risultato e la sua applicazione.

Il controllo in rete di processi si pone proprio in questo scenario, con l'obbiettivo di poter permettere ad un utente situato in una qualsiasi parte del mondo di poter interagire con un determinato macchinario, al fine di ottenere

particolari risultati. Per esempio tale macchinario può consistere in un robot industriale, che deve essere riprogrammato al fine di potergli far compiere nuove operazioni; in questo caso non sarà più necessario che l'equipe di esperti addetta alla riprogrammazione del robot debba muoversi dalla propria sede, bensì potrà operare senza spostarsi, con un grande risparmio di tempo e di risorse finanziarie.
È proprio in questa ottica che si pone il progetto di un tele-laboratorio, ossia di una struttura virtuale che permetta di poter interagire con dei processi fisici. Questo consente infatti all'utente di poter disporre di apparecchiature più o meno sofisticate senza dover sostenere gli oneri dovuti all'acquisto del processo e di tutti gli altri dispositivi necessari per il collegamento con il calcolatore. Viene inoltre risparmiato all'utente il compito di "messa in linea" del processo, ossia di tutte quelle operazioni essenziali per il corretto funzionamento dello stesso (alimentazione, taratura e messa in sicurezza dei dispositivi, interfacciamento con le schede di acquisizione, generazione o adattamento del software, ecc...), che possono risultare forvianti rispetto al compito che l'utente deve svolgere.
Un'altra caratteristica che privilegia i tele-laboratori rispetto ai laboratori tradizionali è quella di poter permettere l'interazione dell'utente anche nel caso in cui il processo sia collocato in siti particolarmente svantaggiosi o nocivi per l'essere umano, quali ad esempio zone montagnose o desertiche difficilmente raggiungibili, postazioni ad elevato rischio di contaminazione batteriologica o nucleare, e così via.
Nonostante che il progetto in esame sia rivolto soprattutto al campo didattico, un generico tele-laboratorio può riguardare i seguenti settori:

- Settore didattico.
- Settore industriale.
- Settore ricerca e sviluppo.

Dal punto di vista didattico, un tele-laboratorio permette di poter interagire con un determinato processo tramite qualsiasi calcolatore collegato ad Internet

ed in qualsiasi orario, ottimizzando al massimo l'utilizzo di tale risorsa anche al di fuori dei laboratori. Inoltre, astraendo dai problemi inerenti la messa in funzione del processo, gli studenti possono dedicarsi al massimo allo studio del controllo dello stesso, progettando nuovi controllori e verificandone la qualità in un contesto che non si riconduce ad una semplice simulazione al calcolatore, bensì ad un vero e proprio caso reale.

Capitolo 1

Problemi di messa in linea di processi

In questo capitolo sono esposti i problemi a cui si deve far fronte al fine di rendere disponibile un processo che possa essere controllato da un utente in laboratorio; non verrà quindi preso in esame il problema del controllo remoto, ma verranno descritte tutte quelle operazioni che dovranno essere eseguite per permettere di interagire con un determinato processo. Tali operazioni saranno effettuate solo per installare il processo e sono rappresentate in figura 1.1.

1.1 Scelta del processo

In questa fase deve essere scelto il tipo di processo che dovrà essere interfacciato con il calcolatore al fine di permetterne il controllo. Vi sono vari tipi di processi fisici che possono essere raggruppati nelle seguenti categorie:

- Processi meccanici.
- Processi idraulici.
- Processi termici.
- Processi elettrici.
- Processi chimici.

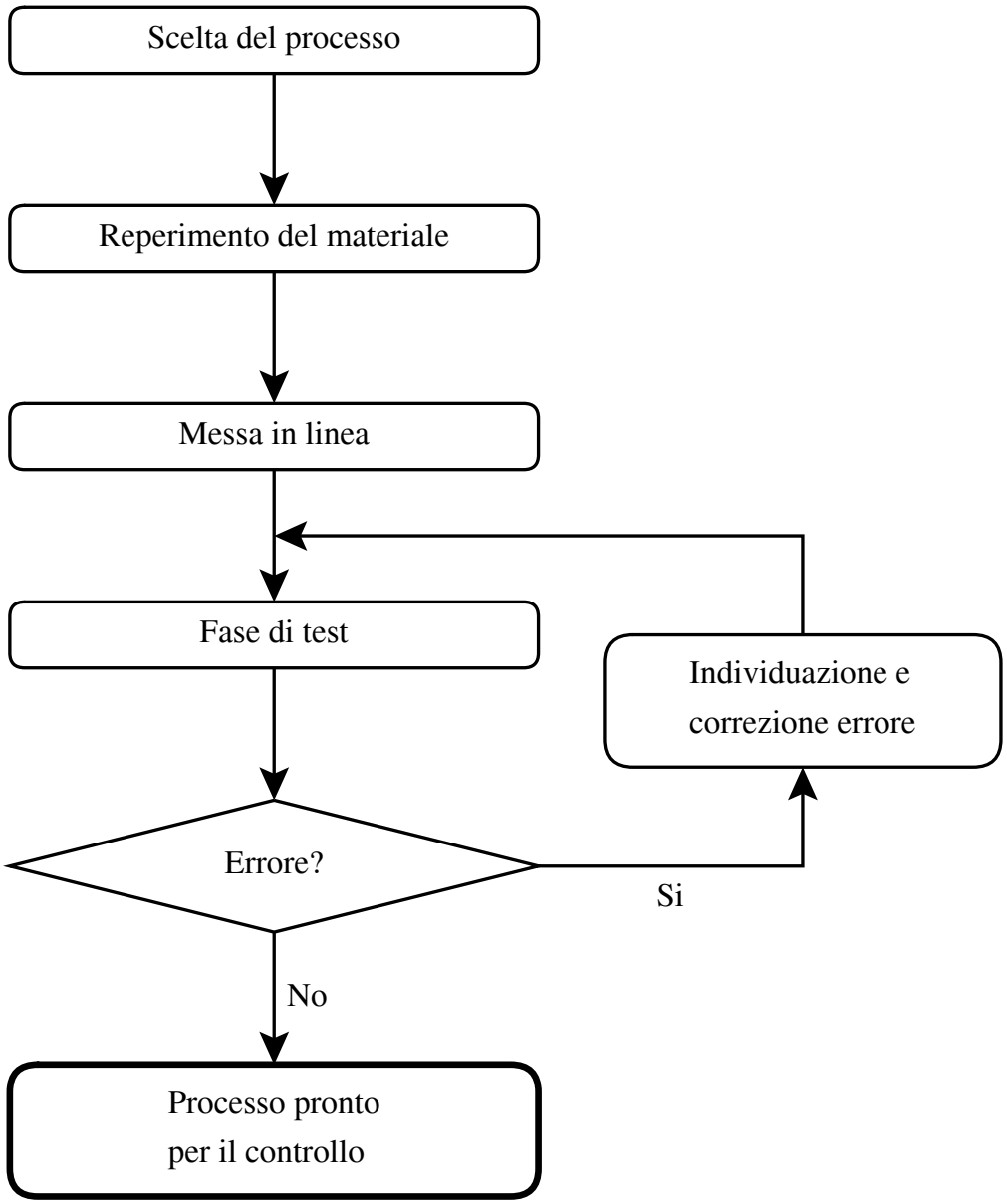

Figura 1.1: Fasi di preparazione del processo.

Naturalmente esistono anche processi più complessi che possono interessare più di una categoria sopraelencata. La scelta del processo dovrà essere motivata dal fine che deve essere perseguito, nonché da altri fattori quali il costo, il consumo, l'ingombro, ecc. Sarà inoltre necessario effettuare uno studio di fattibilità al fine di assicurarsi che tutte le apparecchiature necessarie siano disponibili o reperibili sul mercato.

1.2 Reperimento del materiale necessario

In questa fase dovranno essere scelti ed eventualmente acquistati i componenti più adatti alle esigenze che vogliamo soddisfare.
I dispositivi necessari al funzionamento saranno essenzialmente alimentatori, sensori o trasduttori e attuatori, nonché una o più schede di acquisizione dati analogico-digitale (A/D) e digitale-analogico(D/A), indispensabili per l'interfacciamento con il calcolatore. Quest'ultimo dovrà essere scelto in base al processo, in quanto dovrà avere una potenza di calcolo sufficiente da poter gestire i tempi di campionamento richiesti ed un sistema operativo in grado di garantire il tempo reale; tali tempi di campionamento possono oscillare dall'ordine dei secondi (processi lenti) fino ai millisecondi o meno (processi con costanti di tempo molto basse). All'uscita della scheda D/A dovrà inoltre essere collegata una scheda di amplificazione in grado di fornire una sufficiente potenza agli attuatori del processo.
Una volta reperito tutto il materiale sopra descritto sarà possibile passare alla fase successiva.

1.3 Messa in linea del processo

Con questo termine si intendono tutte quelle operazioni che devono essere effettuate per poter permettere il corretto funzionamento del processo. In generale queste procedure riguardano il collegamento dei dispositivi reperiti nella fase precedente e possono interessare: l'alimentazione degli attuatori e dei trasduttori, l'interfacciamento di questi ultimi con la scheda di acquisizio-

ne, l'applicazione di eventuali meccanismi di sicurezza e la schermatura delle apparecchiature nei confronti di interferenze di origine elettromagnetica. Dovrà inoltre essere implementato anche il software che gestisce la comunicazione della scheda di acquisizione con il programma di controllo vero e proprio.

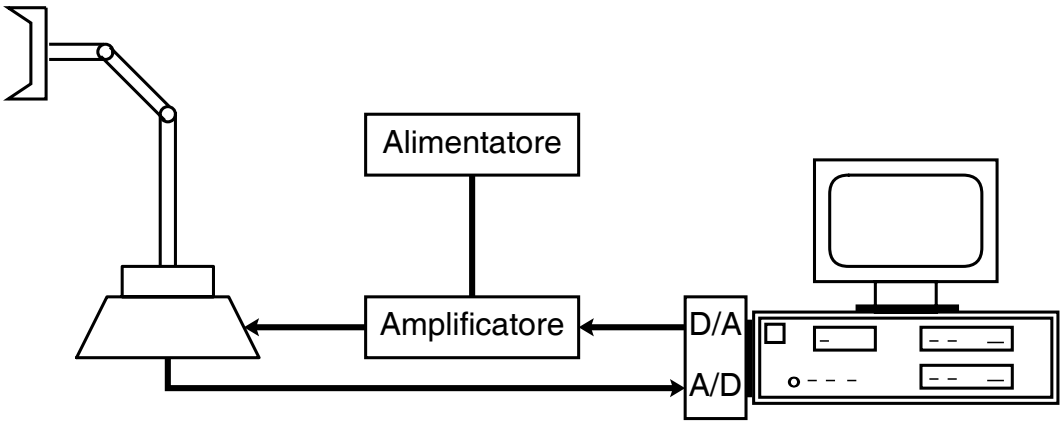

Figura 1.2: Semplice schema inerente il collegamento di un processo.

1.4 Fase di test ed eventuale debugging

Una volta collegate tra di loro tutte le apparecchiature necessarie, è opportuno compiere un'operazione di test, al fine di poter riscontrare eventuali errori commessi nelle fasi precedenti. Sarà quindi controllato il corretto funzionamento dei vari componenti, che, in questa circostanza, possono essere sollecitati fino al loro limite massimo, in modo da garantire, in caso di successo, un certo margine di sicurezza quando essi saranno utilizzati in condizioni meno estreme.
Nel caso insorgesse qualche problema si rende necessaria una fase di debugging, ossia di ricerca dell'errore; non è infatti sempre banale rintracciare l'errore, in quanto esso può essere dovuto a varie cause, come ad esempio ad una non corretta alimentazione, ad interferenze elettromagnetiche, ad una saldatura difettosa od altro. Una volta individuato sarà necessario riparare l'errore, dopodiché si renderà necessaria una nuova operazione di test, fino a che tutto non funzionerà correttamente.

1.5 Procedure inerenti il controllo

A questo punto il processo risulta essere pronto per la fase di controllo. Verranno in seguito esposte tutte quelle operazioni necessarie per il corretto controllo del processo, nonché saranno descritti i problemi a cui bisogna far fronte per non incorrere in difetti di funzionamento. Tale fase comprende diverse operazioni che possono essere riassunte nelle seguenti:

- Analisi e modellazione del processo.
- Scelta del tempo di campionamento.
- Scelta del controllore.

1.5.1 Analisi e modellazione del processo

A questo punto si rende necessario effettuare un'analisi al fine di ottenere un modello matematico che rappresenti il più verosimilmente possibile il processo in questione. Dovremo quindi conoscere tutte le costanti fisiche del sistema, quali lunghezze, masse, coefficienti di attrito, coppie sviluppate, ecc..., che possono essere ricavate dalle tabelle tecniche dei singoli componenti (fornite dal produttore), oppure stimate o misurate sperimentalmente in laboratorio. Una volta ottenuti questi dati è possibile costruire un modello matematico del processo, che verrà poi reso noto all'utente incaricato al controllo dello stesso. Per modello matematico [5] intendiamo un insieme di equazioni e di parametri che ci permettono di determinare gli andamenti nel tempo delle uscite, noti quelli degli ingressi. Nell'ambito dei controlli automatici vengono solitamente utilizzati dei modelli dinamici, preferibili rispetto a quelli statici in quanto questi ultimi non forniscono alcuna informazione riguardo al regime transitorio del processo, ossia sull'andamento nel tempo delle uscite durante il passaggio da uno stato di regime stazionario ad un altro.
È necessario comunque ricordare che il modello ottenuto, per quanto sofisticato esso sia, non potrà mai rappresentare con assoluta esattezza il processo reale, per cui sarà necessario tenere conto di questo in fase di progettazione del controllore.

1.5.2 Scelta del tempo di campionamento

Per tempo di campionamento intendiamo quella frazione di tempo che intercorre tra due successive letture dei trasduttori, e conseguentemente tra due successive elaborazioni del controllore. Tale tempo di campionamento, che in linea teorica dovrebbe essere il minore possibile, in realtà non potrà scendere al di sotto di una certa soglia stabilita dalle limitazioni fisiche delle schede di acquisizione dati e dalla potenza di calcolo dell'elaboratore. È opportuno infatti ricordare che il computer preposto al controllo deve poter effettuare tutti i calcoli necessari entro quel lasso di tempo, pena la perdita del tempo reale. Il tempo di campionamento deve comunque rimanere sufficientemente inferiore alla minima costante di tempo del sistema, affinché non si verifichino perdite o distorsioni di dati inerenti l'andamento reale del processo (aliasing). Nel caso che questa condizione non possa essere verificata (a causa delle limitazioni precedentemente descritte), non sarà possibile poter controllare il processo, a meno che non vengano sostituite le componenti insufficienti con altre più sofisticate (solitamente computers con maggiore potenza di calcolo).

1.5.3 Tecniche di controllo

Per quanto riguarda le tecniche di controllo, queste si suddividono tra quelle in azione diretta (catena aperta) e quelle in retroazione (catena chiusa).

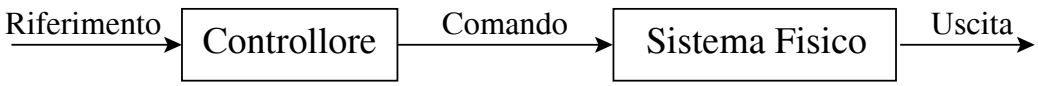

Figura 1.3: Semplice controllo ad azione diretta.

In un controllo ad azione diretta (Fig. 1.3) il valore della variabile manipolabile non dipende da quello della variabile controllata né da quelli di altre variabili dipendenti del sistema controllato; tutto questo comporta che il valore dell'ingresso venga determinato all'interno del regolatore in base ad un modello matematico, senza operare alcuna verifica sulla rispondenza del valo-

re della variabile controllata. Per questo motivo vengono per lo più privilegiati i controllori che contengono almeno un percorso di segnale chiuso (anello di retroazione), in cui il valore della variabile manipolabile viene determinato in base alla misura della variabile controllata (Fig. 1.4).

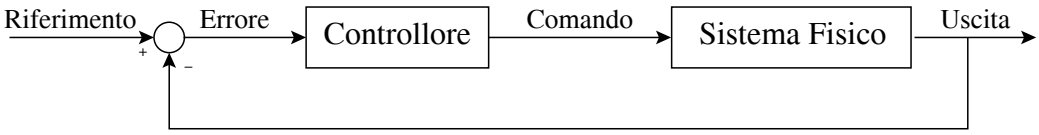

Figura 1.4: Semplice controllo in retroazione.

In questo caso saranno naturalmente necessari uno o più trasduttori per effettuare la misura dell'uscita. In modo maggiormente dettagliato un controllo in catena chiusa può essere schematizzato come in figura 1.5.

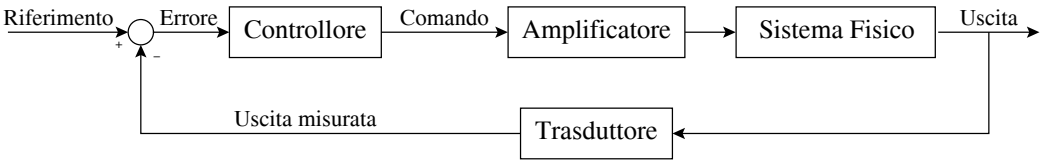

Figura 1.5: Controllo in retroazione.

Dato un sistema fisico esistono diverse tecniche di per progettare un regolatore che permetta il corretto controllo del processo. Tali tecniche si distinguono in base al fatto che il sistema sia lineare o meno, che il modello ad esso associato sia più o meno fedele, oppure a seconda delle prestazioni che desideriamo ottenere. In ogni caso la progettazione di un controllore (specie per processi non banali) richiede una certa esperienza, nonché una buona conoscenza della materia in questione.

Capitolo 2

Specifiche del progetto e tecnologia Internet

In questo capitolo verranno esposte le principali caratteristiche inerenti il progetto in questione, quali l'organizzazione, gli obbiettivi prefissati ed il confronto con altri laboratori virtuali esistenti. Saranno inoltre descritti i concetti di base della rete Internet e del linguaggio di programmazione Java.

2.1 Caratteristiche del Tele-Laboratorio

Nel precedente capitolo sono state esposte tutte le operazioni necessarie a far funzionare un determinato processo in locale, ossia da una postazione situata nei pressi del processo stesso; una volta completate tali operazioni sarà possibile azionare l'esperimento, ma l'operatore dovrà necessariamente trovarsi nella medesima sede del processo. Tutto questo potrà essere evitato grazie alla telematica ed alla rete Internet, che consente di poter collegare e far comunicare tra di loro due o più elaboratori indipendentemente dalla loro ubicazione territoriale. L'utente può quindi collegarsi alla rete e raggiungere il sito del tele-laboratorio (Fig. 2.1) tramite un programma di navigazione (browser).
A questo punto si troverà di fronte alla pagina iniziale, dalla quale, mediante comandi estremamente intuitivi, avrà la possibilità di effettuare le seguenti scelte:

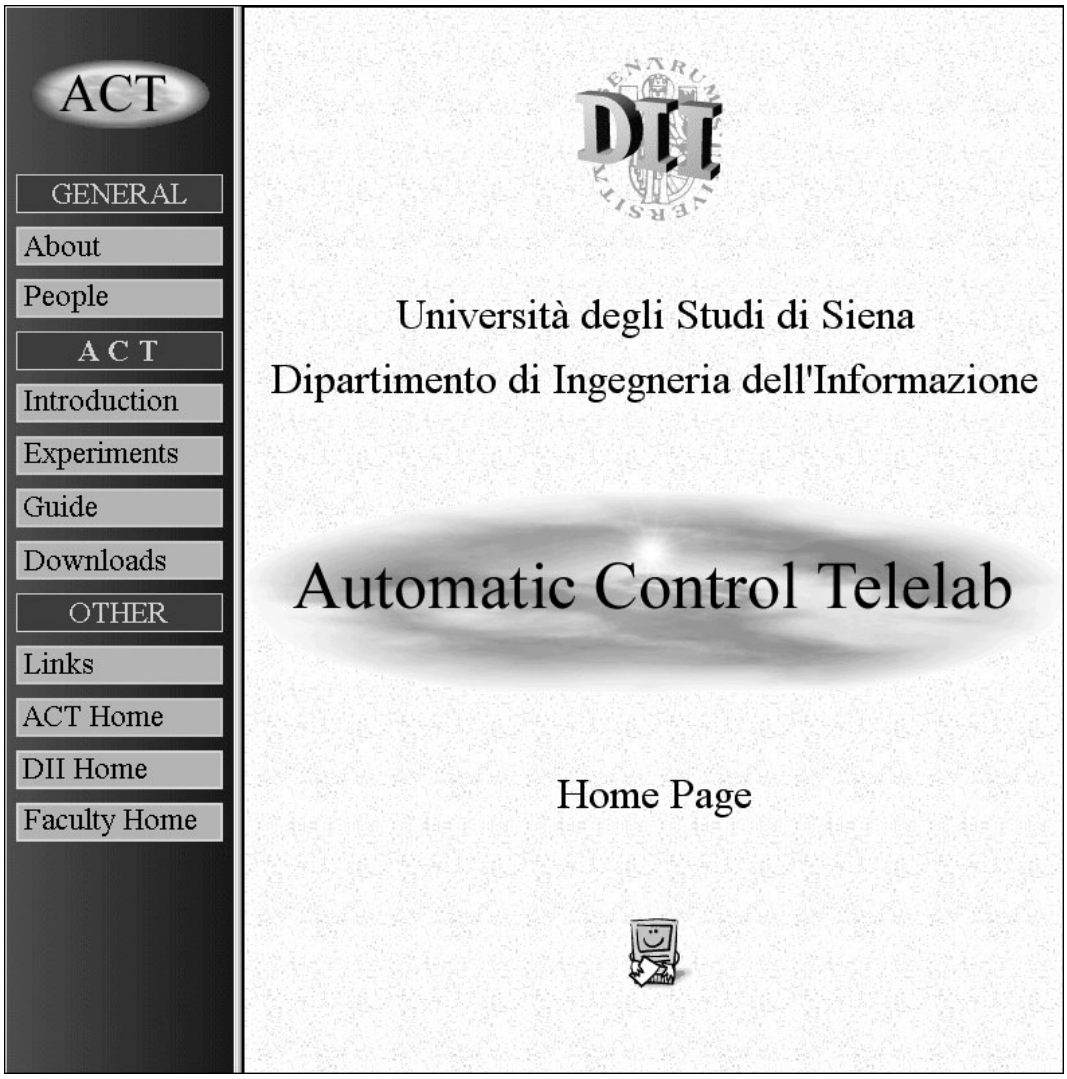

Figura 2.1: Home page del Tele-Laboratorio.

- Visualizzazione di una descrizione introduttiva.
- Visualizzazione del manuale d'uso.
- Visualizzazione dello staff di sviluppo.
- Visualizzazione della lista dei processi disponibili.

Per quanto concerne il manuale d'uso, è bene specificare che esso riguarda le operazioni che possono essere compiute dall'utente finale, e non interessa minimamente tutte le procedure di installazione software e hardware necessarie per il corretto funzionamento del tele-laboratorio, le quali saranno effettuate sul luogo del processo da personale qualificato.
Nel caso che l'utente decida di proseguire con la gestione di un esperimento in remoto, si troverà di fronte alla lista dei processi disponibili per il controllo; tali processi possono essere di vario tipo, da quelli meccanici a quelli elettrici, da quelli idraulici a quelli chimici, e così via.
Una volta effettuata la scelta del processo, l'utente potrà disporre di una rappresentazione dettagliata dello stesso, in cui, oltre ad informazioni di carattere generale, sarà descritto un modello matematico che lo rappresenta. Nel caso che tale modello non soddisfi le esigenze dell'operatore, quest'ultimo potrà costruirselo uno personalizzato, in quanto saranno messe a disposizione anche tutte le caratteristiche fisiche del sistema, quali lunghezze, masse, ecc.
Proseguendo ancora verranno elencati i controllori predefiniti (Fig. 2.2), ossia quei controllori che sono stati progettati e verificati in locale. Nel caso che l'utente desideri comunque realizzare una propria legge di controllo, questo sarà possibile purché vengano rispettate delle convenzioni che saranno esposte con maggior rigore nei prossimi capitoli. In ogni caso il controllore dovrà essere un file nel formato Simulink, un toolbox facente parte del pacchetto applicativo Matlab. In questo caso il regolatore progettato dall'utente verrà inviato via Internet fino all'elaboratore preposto alla gestione del processo, dopodiché verrà verificato che il file sia corretto, ossia che risponda alle convenzioni imposte.
A questo punto si può procedere alla fase di esecuzione (Fig. 2.3) dalla quale sarà possibile avviare il processo; potrà inoltre essere modificato il riferimento

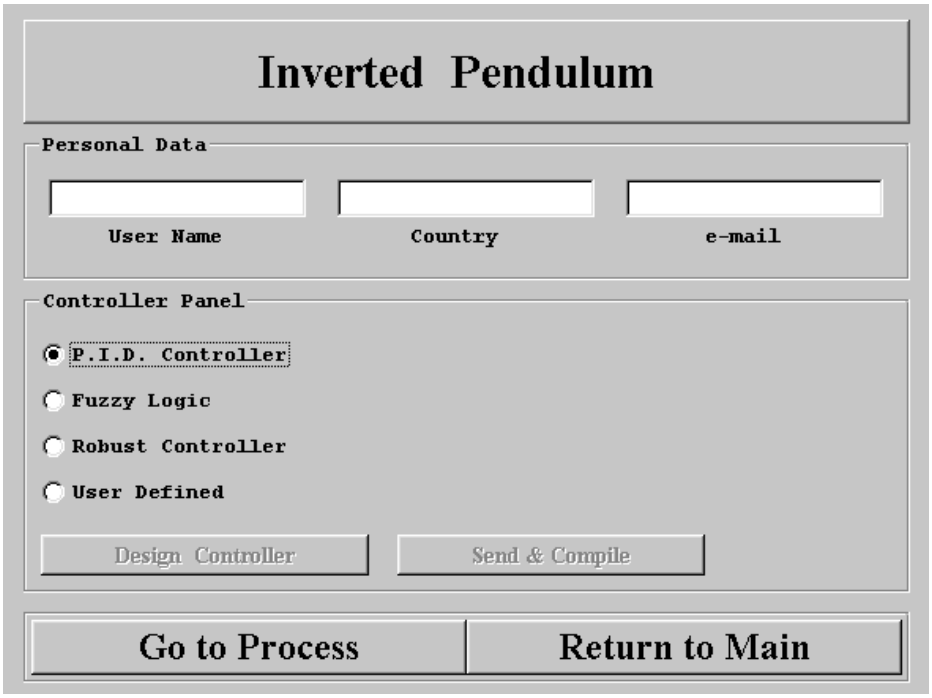

Figura 2.2: Pagina relativa alla scelta del controllore.

in fase di esecuzione, nonché variare alcuni parametri relativi al controllore, sempre in tempo reale. L'andamento del processo verrà visualizzato in opportuni grafici che rappresentano il riferimento, l'uscita e il comando fornito al processo. Sarà inoltre disponibile un feedback di tipo visivo, realizzato mediante un'opportuna telecamera, che consentirà di verificare visivamente il processo, come se ci trovassimo sul luogo dello stesso (Fig. 2.4).

Al termine dell'esperimento sarà possibile recuperare il dettaglio temporale delle variabili in questione, memorizzate in formato .MAT usato da Matlab; questo consentirà di poter effettuare delle analisi più rigorose riguardo l'andamento del processo in esame.

È utile ricordare che è possibile porre delle limitazioni di accesso al telelaboratorio in base al valore immesso nei campi relativi ai dati personali dell'utente (Fig. 2.2). Questo potrà permetterne l'uso esclusivo da parte di utenti facenti parte di una determinata università od organizzazione.

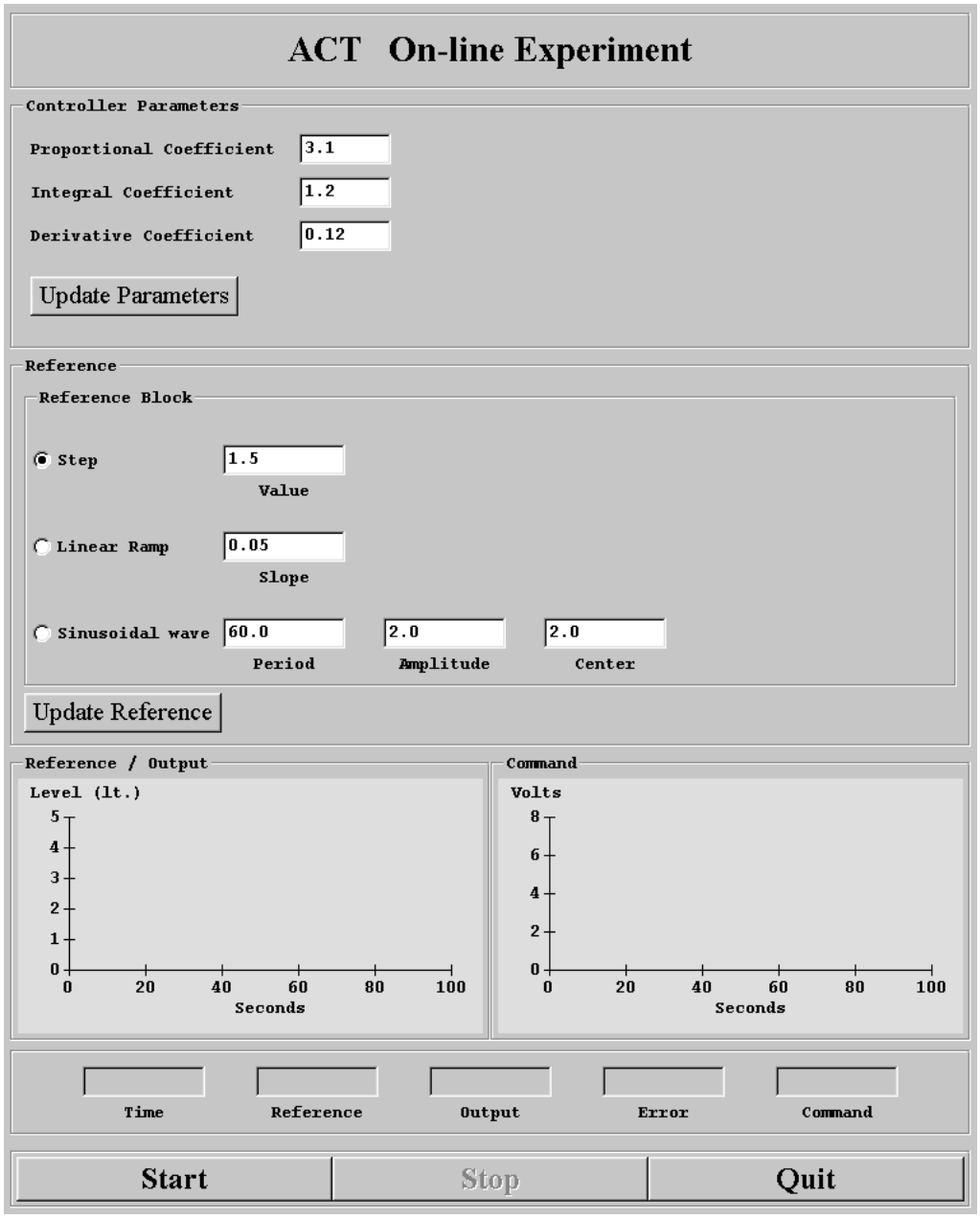

Figura 2.3: Pagina relativa alla gestione del processo.

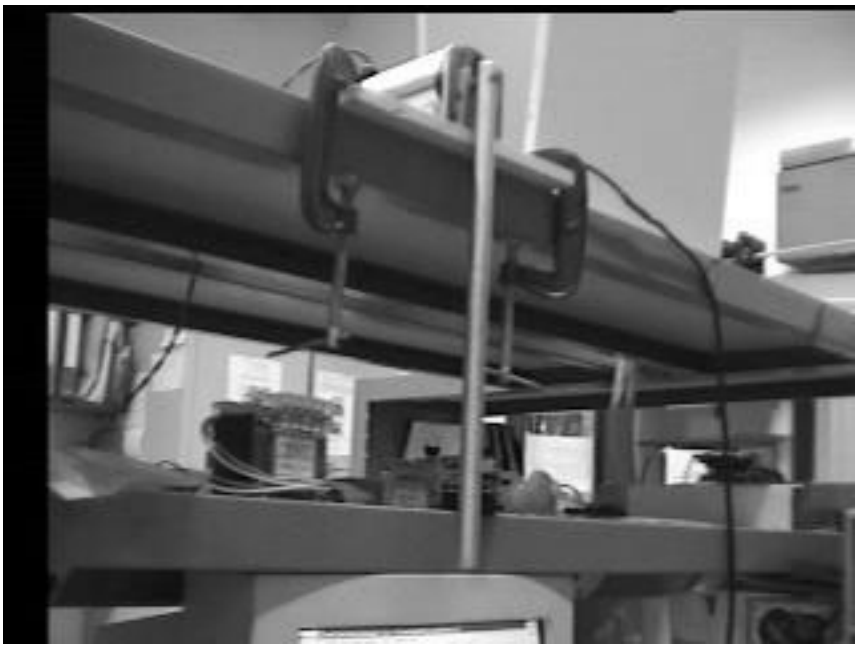

Figura 2.4: Visualizzazione del processo mediante telecamera.

2.2 Confronto con altri laboratori virtuali

Il termine "laboratorio virtuale", coniato in questi ultimi anni con l'espansione dell'informatica, non ha ancora assunto un significato ben preciso. È infatti possibile che esso riguardi un laboratorio telematico simile a quello precedentemente descritto, oppure un'applicazione software che non abbia alcun legame con dei processi fisici. In questo caso il processo sarà simulato tramite opportuni programmi che terranno conto delle varie leggi della fisica al fine di poter riprodurre il più fedelmente possibile la realtà (Fig. 2.5). Sono proprio questi tipi di laboratori ad essere maggiormente diffusi nel mondo, probabilmente a causa del minor costo che essi richiedono (assenza di tutte le componenti hardware relative ai processi). Gli altri vantaggi che gli sviluppatori di questi laboratori indicano nei confronti dei laboratori tradizionali sono l'assenza dei dispositivi di sicurezza e dei conseguenti rischi che un processo reale può comportare, nonché la possibilità di condividere simultaneamente l'esperimento con un grande numero di utenti; inoltre tali utenti possono fare quello che vogliono, senza dover temere rotture o malfunzionamenti dei meccanismi.

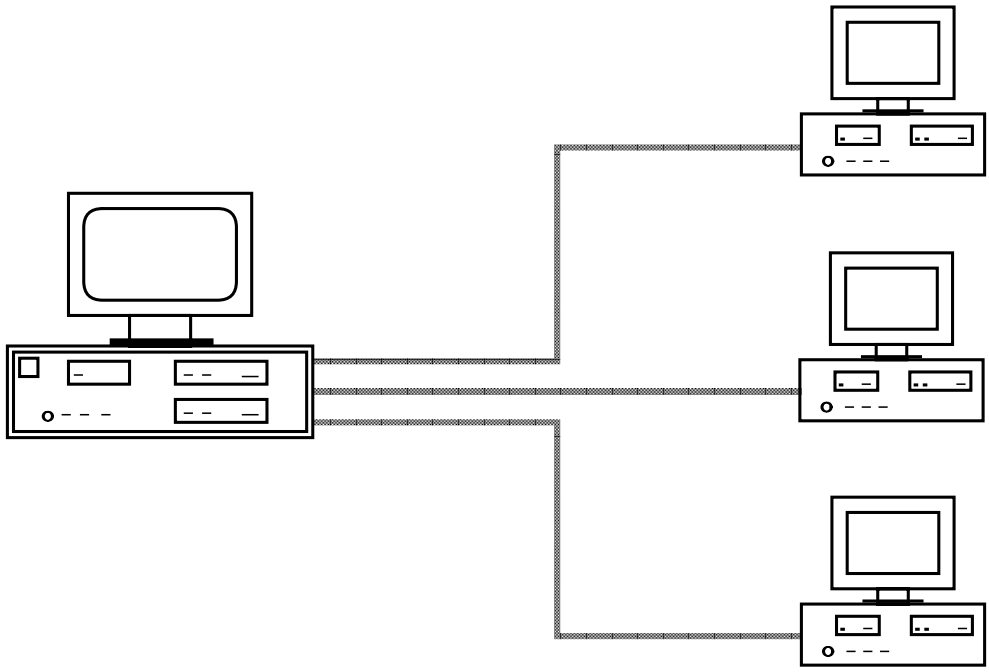

Figura 2.5: Schema di laboratorio virtuale con simulazione software.

Per quanto concerne i laboratori che fanno uso di processi fisici (Fig. 2.6), questi hanno l'indubbio vantaggio di proporre all'utente una situazione reale e non una simulazione al calcolatore, che, per quanto sofisticata essa sia, non potrà mai rappresentare tutti gli aspetti inerenti un processo concreto. Inoltre un caso reale comporta anche uno studio relativo alla modellizzazione del processo, fase di fondamentale importanza per il futuro controllo dello stesso; tutto questo viene invece a mancare nel caso della simulazione software in cui il modello viene fornito con certezza.
Oltre a questo, il presente progetto si contraddistingue rispetto alla maggior parte degli altri laboratori simili per almeno quattro caratteristiche:

- Possibilità di variare il riferimento in tempo reale.
- Possibilità di variare alcuni parametri del controllore in tempo reale.
- Possibilità di progettare in remoto il controllore.
- Possibilità di una visualizzazione mediante telecamera.

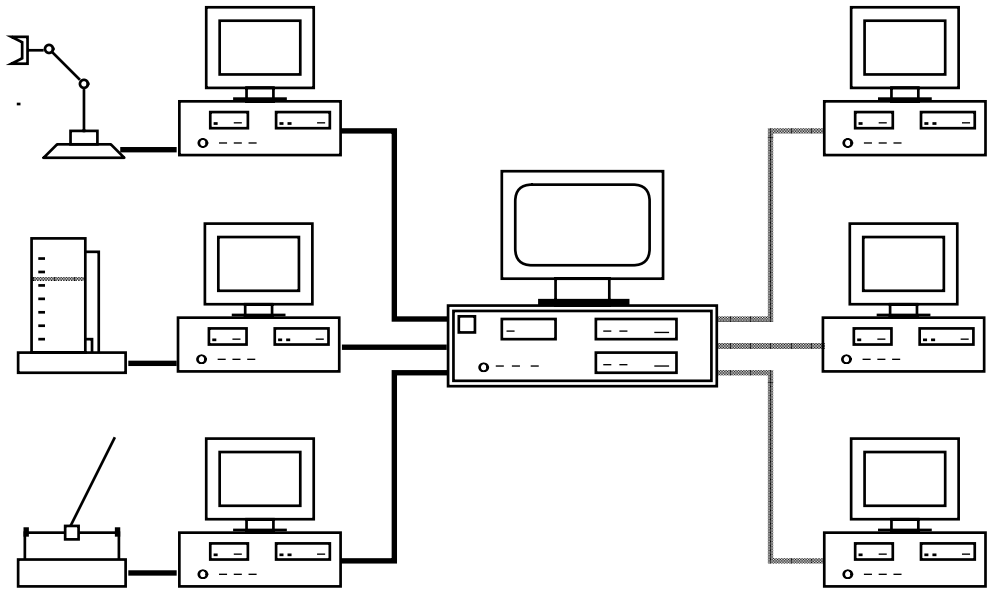

Figura 2.6: Schema di laboratorio virtuale con processi fisici.

Infatti, in quasi tutti i tele-laboratori, è necessario definire a priori il segnale di riferimento che si desidera applicare, dopodiché non sarà più possibile modificarlo; lo stesso vale per quanto riguarda la variazione di alcuni parametri del regolatore, che possono talvolta essere modificati solamente prima che l'esperimento abbia inizio. Per quanto riguarda la visualizzazione del processo mediante telecamera, bisogna ricordare che la fluidità dalle immagini dipenderà notevolmente dalla distanza e dal sovraccarico che sarà presente sulla linea trasmissiva al momento dell'esperimento.

2.3 Fondamenti della rete Internet

L'idea di collegare tra di loro vari computers dislocati non nello stesso edificio, ma a migliaia di chilometri di distanza, nacque alla fine degli anni '60 negli Stati Uniti d'America [8]. Tale progetto di ricerca fu finanziato dal governo locale e la sua applicazione fu diretta soprattutto all'ambito militare. Negli anni '90 tale rete fu convertita ad uso civile, permettendo a chiunque di potervi accedere; è proprio in questo periodo che fu coniato il termine "world wide Internet" o più semplicemente Internet.

Una rete di così grandi proporzioni, con milioni di calcolatori che ne fanno parte, necessita di sofisticati meccanismi di comunicazione che comtemplino ogni possibilità; niente deve essere lasciato al caso. Tali meccanismi sono chiamati protocolli di comunicazione e possono essere a loro volta suddivisi in vari strati (layer).

2.4 Il protocollo TCP/IP

In una connessione possiamo distinguere i due elaboratori in "client" e "server". Per convenzione il server è colui che dispone di una certa risorsa per cui ne viene fatta richiesta da parte del client. I dati relativi ad una richiesta che vengono trasmessi lungo la rete sono suddivisi in pacchetti, ossia in piccoli contenitori che una volta arrivati a destinazione vengono ricomposti fino a riottenere il messaggio originale. Tale processo viene fornito dal protocollo TCP/IP, il quale è suddiviso nei seguenti quattro strati (Fig. 2.7):

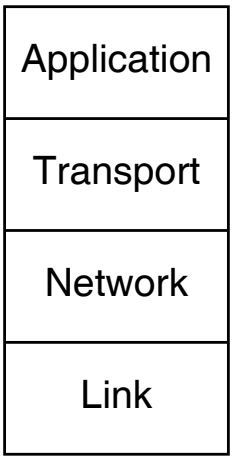

Figura 2.7: I quattro strati del protocollo TCP/IP.

1. Link layer. Questo livello si occupa dell'interfacciamento fisico dell'elaboratore alla rete, a seconda della scheda di rete utilizzata e dal sistema operativo in uso. Esso deve infatti risolvere tutti i problemi relativi all'interfacciamento con il cavo da parte dell'hardware presente. Inter-

net è infatti totalmente indipendente dalla piattaforma utilizzata proprio grazie a questo layer, che si preoccupa di ricondurre le varie tecnologie costruttive ad uno standard comune.

2. Network layer. Questo livello si occupa del movimento dei pacchetti attraverso la rete, instradandoli verso le direzioni più opportune (routing). Tale operazione è piuttosto complessa, in quanto esistono varie linee che collegano due computers, alcune delle quali possono essere interrotte senza preavviso a causa di sovraffollamenti; sarà quindi compito di questo layer smistare opportunamente il traffico verso altre linee meno intasate.

3. Transport layer. Poiché i pacchetti possono seguire strade diverse, con molta probabilità essi arriveranno a destinazione in modo disordinato; è proprio il compito di questo livello riordinare tutti i pacchetti arrivati, nonché segnalare eventuali pacchetti che mancano all'appello; in questo caso tali pacchetti verranno trasmessi nuovamente dalla sorgente, fino a che il flusso dei dati non è completato.
   In casi particolari si può fare uso anche di una trasmissione inaffidabile, ossia che non si preoccupa dell'ordine o del mancato recapito di alcuni pacchetti; questo viene utilizzato ad esempio per la trasmissione della voce, in cui l'utilizzo del protocollo più affidabile comporterebbe ingenti ritardi che impedirebbero una corretta comunicazione.

4. Application layer. Questo strato si occupa di gestire i dettagli della particolare applicazione che stiamo eseguendo. Le principali procedure che la maggior parte delle applicazioni utilizza, sono:

   - Telnet, un terminale virtuale che consente l'utilizzo remoto di un calcolatore.
   - FTP (File Transfer Protocol), che consente lo scambio dei file a distanza.
   - SMTP (Simple Mail Transfer Protocol), utilizzato per lo scambio di messaggi tramite posta elettronica.

2.5 Indirizzi di Internet

In una struttura così estesa che comprende milioni di calcolatori è necessario che ad ognuno di essi venga associato un indirizzo, ossia un identificatore che lo distingua in modo univoco da tutti gli altri. Il formato che Internet utilizza per rappresentare gli indirizzi è un numero a 32 bit, che viene solitamente scritto come una sequenza di 4 numeri (ad 8 bit) separati da un punto. Un esempio di indirizzo può essere 140.252.13.33.
Nonostante che ogni computer possa essere distinto da un altro grazie a questo indirizzo, esiste un metodo alternativo per individuare un determinato calcolatore; tale metodo è chiamato DNS (Domain Name System) e consente di associare un nome mnemonico ad un certo indirizzo. Quando un utente si collegherà alla rete alla ricerca di un certo sito potrà quindi inserire il DNS del sito, che risulterà senz'altro più agevole da ricordare.

2.6 Internet e Java

La maggiore funzionalità di Internet che viene utilizzata è quella della "navigazione", ossia della possibilità di visitare siti dislocati in tutto il mondo; ogni sito mette a disposizione varie pagine in formato HTML (Hyper Text Markup Language), che possono includere testi, immagini e collegamenti con altre pagine. Fino a pochi anni fa non era però possibile eseguire in rete dei programmi, vista la varietà di calcolatori e di sistemi operativi presenti. Per questo motivo è stato realizzato Java, un linguaggio di programmazione orientato agli oggetti per Internet, che consente di eseguire dei programmi (applets) durante la navigazione indipendentemente dal tipo di elaboratore e dalla piattaforma usata.
Le caratteristiche che contraddistinguono Java dagli altri linguaggi di programmazione sono essenzialmente due:

- Indipendenza dalla piattaforma di sviluppo.
- Presenza di meccanismi di sicurezza nei confronti di altri utenti (firewall).

Questa seconda caratteristica è di fondamentale importanza perché protegge l'utente da eventuali accessi non autorizzati al disco fisso o alla memoria. L'applet potrà quindi influenzare solamente la parte di memoria che gli viene riservata e che sarà automaticamente ripulita al termine della stessa.

Capitolo 3

Architettura del Tele-Laboratorio

In questo capitolo sarà presentata l'architettura di base del progetto, analizzando distintamente tutte le sue componenti, che verranno poi descritte in modo più dettagliato nel prossimo capitolo.

3.1 Architettura generale del progetto

L'intero progetto può essere diviso in due parti sufficientemente distinte tra di loro, che riguardano il lato client, ossia la parte relativa all'utente remoto, ed il lato server, quello preposto all'interfacciamento con il processo.
Per quanto concerne il lato client, tutto il software è stato realizzato mediante pagine HTML ed applets Java. Queste scelte sono state motivate dall'intenzione di rendere il più possibile 'amichevole' (friendly) l'interfaccia utente, nonché permettere l'accesso al tele-laboratorio indipendentemente dalla piattaforma e dal browser adottato per la navigazione. Le pagine HTML sono usate per la parte introduttiva, mentre per il controllo vero e proprio vengono utilizzate due applets Java: la prima (Telelab1) si occupa della scelta della legge di controllo, mentre la seconda (Telelab2) della gestione del processo in tempo reale.
Per quanto riguarda il lato server, questo è notevolmente più complicato del precedente sia per quanto concerne l'aspetto hardware che quello software.

Mentre sulla macchina remota non sono presenti componenti hardware aggiuntive, così non è su quella locale, che contiene tutti gli elementi necessari al corretto funzionamento del processo fisico.
Dal punto di vista software è stato utilizzato il sistema operativo Windows 95/98 della Microsoft, che a tutt'oggi è senz'altro il più diffuso al mondo. Per quanto riguarda l'aspetto applicativo, sarà presente un programma sempre in esecuzione che avrà il compito di restare in attesa (listening) di una connessione TCP da parte di un calcolatore remoto. Tale programma è stato sviluppato in C++, un linguaggio di programmazione orientato agli oggetti, ed è stato nominato "Interface", in quanto funge da interfaccia tra l'utente ed il processo fisico, dovendosi occupare di tutte le fasi di connessione e gestione dei controllori.
Per quanto concerne il programma relativo alla gestione del processo in tempo reale, esso è senz'altro la parte più complessa del progetto, in quanto deve provvedere, oltre all'interfacciamento con la scheda di acquisizione, anche alla gestione della variazione del riferimento e dei parametri del controllore, il tutto in tempo reale; anche in questo caso deve essere gestita una connessione TCP con il lato client. Questo programma è stato scritto in C, mediante l'utilizzo di un toolbox di Matlab chiamato Real-Time Workshop (RTW) [3] e di procedure particolari per la gestione del tempo reale e dello scambio di dati. È opportuno ricordare che questo programma, denominato "Process", viene eseguito soltanto al momento in cui l'utente ha deciso il tipo di legge di controllo da utilizzare, in quanto esso varia in funzione del controllore selezionato; nel caso in cui il regolatore sia progettato dall'utente è necessaria una previa fase di compilazione al fine di poter disporre del file eseguibile (Process.exe) sul server di processo.
Per l'efficiente funzionamento della telecamera viene invece utilizzato un pacchetto software appositamente predisposto per teleconferenze via Internet, sia dal lato server che client. Per poter permettere la visualizzazione del processo in tempo reale è però opportuno disporre di un calcolatore dedicato alla gestione della telecamera, poiché la compressione delle immagini richiederebbe un grande utilizzo di risorse del sistema, impedendo il controllo in tempo reale

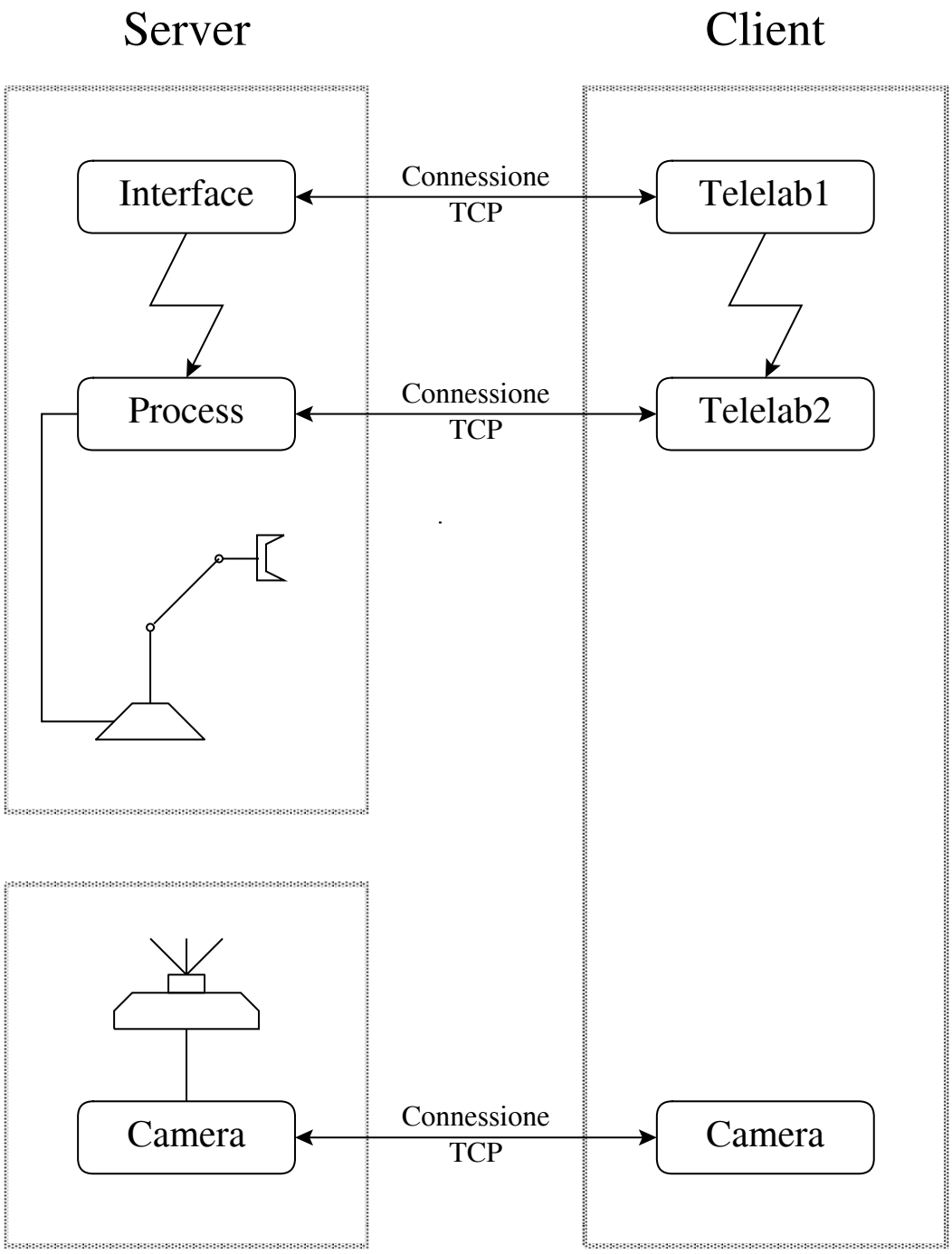

Figura 3.1: Architettura generale del Tele-Laboratorio.

del processo. È comunque opportuno ricordare che il tele-laboratorio è perfettamente funzionante anche in assenza di feedback visivo (l'andamento del processo è espresso tramite opportuni grafici), per cui sarà facoltà dell'utente decidere se attivare o meno questa caratteristica.
Lo schema funzionale dell'architettura client/server è riportato in figura 3.1.

3.2 Descrizione del modulo Interface

Come esposto nel paragrafo precedente, Interface è un programma che rimane sempre in esecuzione sulla macchina locale ed il cui compito è quello di restare in attesa di eventuali connessioni; la porta locale prefissata per tale operazione è la numero 3000.
Una volta stabilita la connessione (con l'applet Telelab1) verranno inviati i seguenti dati: titolo del processo, numero di controllori predefiniti e rispettivi nomi. Tali informazioni saranno state memorizzate nel file "Process.dat", un file di testo che dovrà essere aggiornato ogni volta che verrà implementata una nuova legge di controllo dall'amministratore del tele-laboratorio.
A questo punto viene attivato un timer con un valore di time-out prefissato (es. 5 minuti); se entro tale tempo l'utente non sarà passato alla fase di gestione del processo, la connessione verrà interrotta mediante una chiamata ad una procedura asincrona, al fine di poter rendere possibile l'accesso al tele-laboratorio ad altri utenti. È infatti opportuno ricordare che essendo unica la risorsa, ossia il processo fisico, non ci potrà mai essere collegato più di un utente alla volta.
Nel caso che il time-out non sia scaduto, il programma resta in attesa di ricevere dei dati da parte dell'applet Java; tali dati variano a seconda che l'utente abbia deciso di progettare una propria legge di controllo oppure di utilizzare un controllore predefinito. Nel primo caso sarà ricevuto un pacchetto contenente un opportuno codice che predisporrà il programma alla ricezione ed alla scrittura del file contenente il controllore (Control.mdl); una volta completata la trasmissione, tale file verrà opportunamente inserito all'interno del modello Simulink rappresentante il processo in assenza di controllore (Source.mdl) e

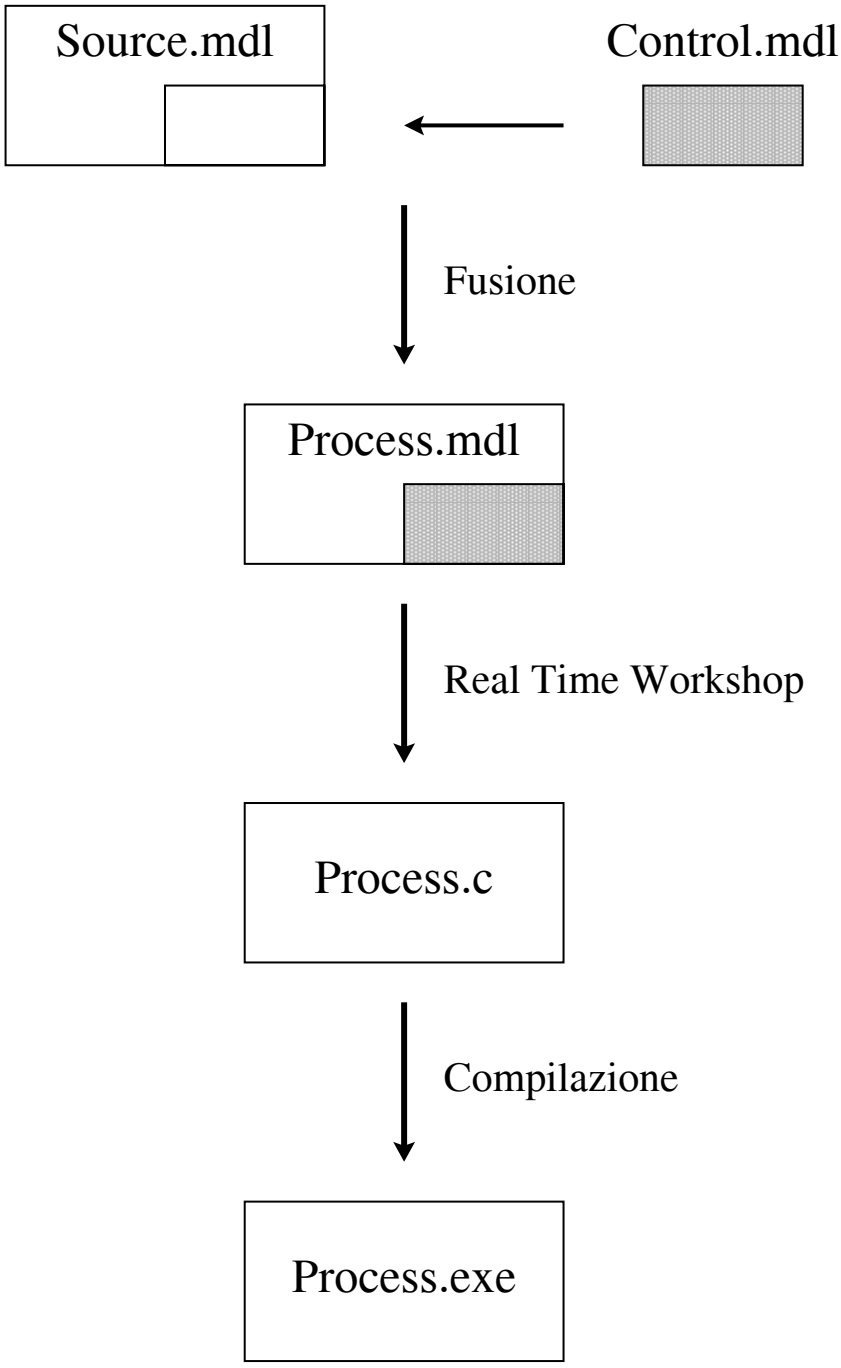

Figura 3.2: Fasi necessarie per ottenere il file eseguibile preposto al controllo del processo.

verrà avviata la procedura di compilazione fornita dal Real-Time Workshop (Fig. 3.2). Al termine di tale operazione verrà spedito il risultato, che nel caso risulti essere positivo, permetterà di proseguire esattamente allo stesso modo di come avremmo fatto utilizzando un regolatore predefinito.
Nel momento in cui l'utente decide di passare alla fase di gestione del processo, verranno ricevuti i dati relativi alle informazioni personali, che saranno verificati e la cui risposta sarà inviata all'applet. Se l'utente risulta tra quelli abilitati verrà ricevuto il codice del controllore scelto, dopodiché sarà eseguito il relativo file di gestione (Process.exe), al termine del quale verrà interrotta la connessione. Il programma si è quindi riportato alle condizioni iniziali, ed è ora in attesa di ricevere nuove connessioni (Fig. 3.3).

3.3 Descrizione dell'applet Telelab1

L'applet Telelab1 è preposta al dialogo con il programma Interface, come rappresentato in figura 3.1, e può essere eseguita da un qualsiasi browser che supporti la versione 1.1 delle JDK (Java Development Kit), un ambiente di sviluppo contenente un insieme di classi standard di Java.
Filosofia generale di ogni applet è quella di essere orientata alla gestione degli eventi, ossia il gestore dell'applet esegue degli opportuni metodi solo in seguito al verificarsi di determinati eventi, che vengono solitamente generati dal mouse, in quanto è sempre presente un'interfaccia visuale per l'utente.
Questa applet (Fig. 3.4) contiene una parte relativa all'inserimento dei dati personali dell'utente, necessari per l'identificazione dello stesso, ed una parte che permette la scelta della legge di controllo da utilizzare. Saranno visualizzati in questa sede i controllori predefiniti e vi sarà la possibilità di progettarne uno personalizzato; in questo caso sarà necessario scrivere in un'apposita finestra il file in formato Simulink preposto al controllo. Per facilitare questa operazione si suggerisce di utilizzare le funzioni "copia" ed "incolla" fornite dal sistema. Una volta disegnato il controllore, dovrà essere spedito mediante il pulsante "Send & Compile".
Ogni eventuale errore dovuto ad una caduta della connessione, ad una sca-

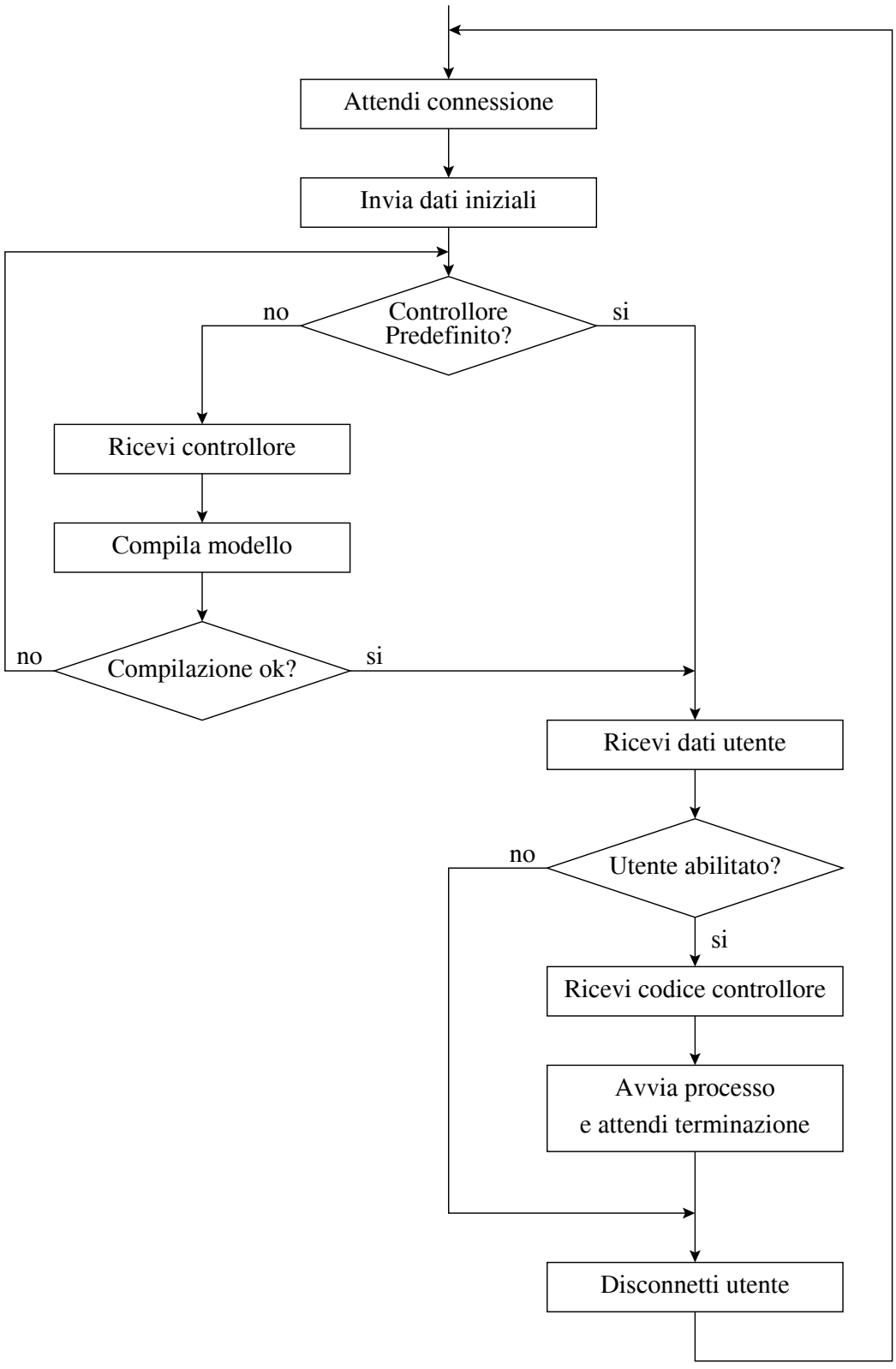

Figura 3.3: Diagramma di flusso relativo al programma Interface.

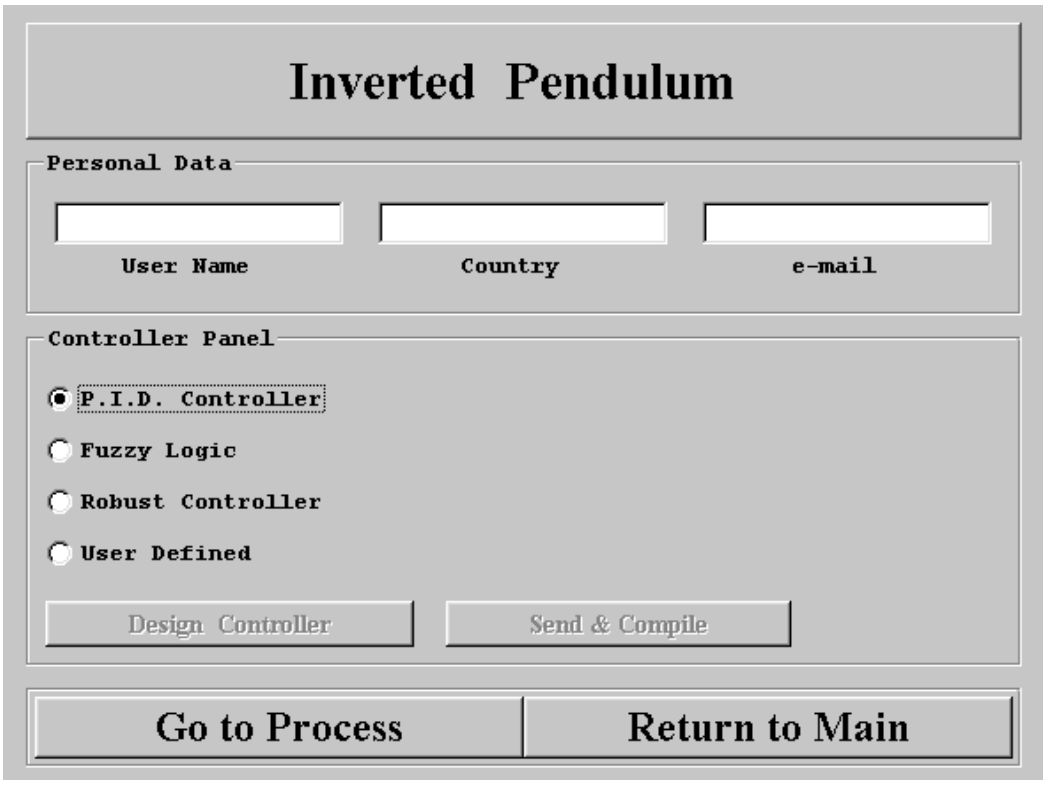

Figura 3.4: Interfaccia visuale dell'applet Telelab1.

denza del time-out o ad un errore di compilazione, sarà visualizzato mediante un'opportuna finestra.

A questo punto l'utente può scegliere se proseguire o tornare indietro alla pagina principale; nel primo caso l'applet viene fermata e viene caricata quella relativa alla gestione del processo, mentre nel secondo viene visualizzata la pagina HTML principale del tele-laboratorio.

Per quanto riguarda l'aspetto relativo alla connessione TCP con il modulo Interface, essa funziona in modo complementare rispetto a quanto descritto nel paragrafo precedente, ovvero se Interface trasmette dei dati, Telelab1 sarà predisposto a riceverli, e viceversa.

3.4 Descrizione del modulo Process

Una volta deciso il controllore da utilizzare si passa alla fase di gestione del processo, che viene realizzata sul server per mezzo del programma Process. Tale programma viene eseguito da Interface una volta selezionata la legge di controllo da utilizzare; vi sarà un eseguibile già pronto per ogni controllore predefinito.
Poiché il programma non è stato scritto direttamente in C, ma mediante l'ausilio del Real-Time Workshop (RTW), la trattazione verrà divisa in due parti, una relativa alla creazione dell'eseguibile ed una relativa al funzionamento del programma.

3.4.1 Creazione del file Process.exe

Il Real-Time Workshop è un toolbox di Matlab che consente di trasformare un modello Simulink (Fig. 3.5) in un file sorgente scritto in linguaggio C, che, successivamente compilato, consente di poter effettuare simulazioni in modo estremamente più veloce rispetto a quelle eseguite in ambiente Matlab. Aggiungendo al modello Simulink degli opportuni blocchi per l'interfacciamento con la scheda di acquisizione, diventa possibile pilotare il processo. Tali blocchi devono essere realizzati mediante S-Functions, che sono particolari funzioni di Matlab scritte in linguaggio C.

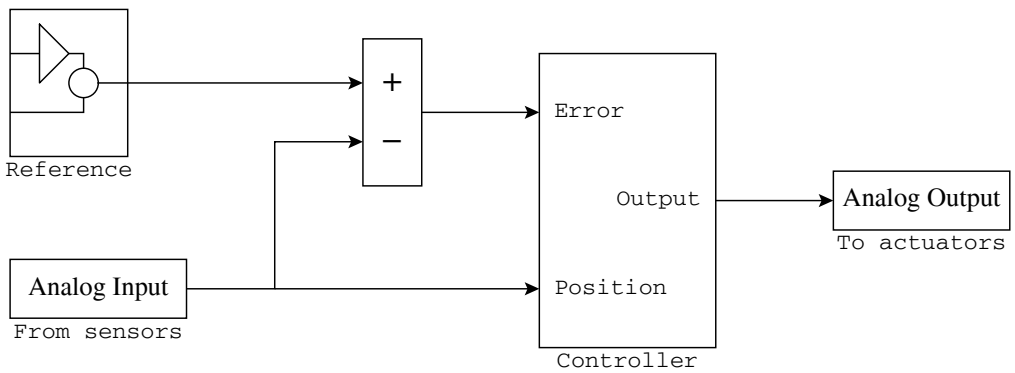

Figura 3.5: Esempio di modello Simulink rappresentante un processo reale.

L'eseguibile ottenuto in questa fase non sarebbe però sufficiente a realizzare tutte le operazioni necessarie, ed in modo particolare conterrebbe le seguenti limitazioni:

- Assenza di connessioni TCP.
- Impossibilità di poter variare il riferimento ed i parametri del controllore in fase di esecuzione.
- Assenza del tempo reale.

Queste caratteristiche sono state realizzate inserendo all'interno dei file sorgenti ottenuti dal Real-Time Workshop opportune procedure preposte a tale scopo (Fig. 3.6).

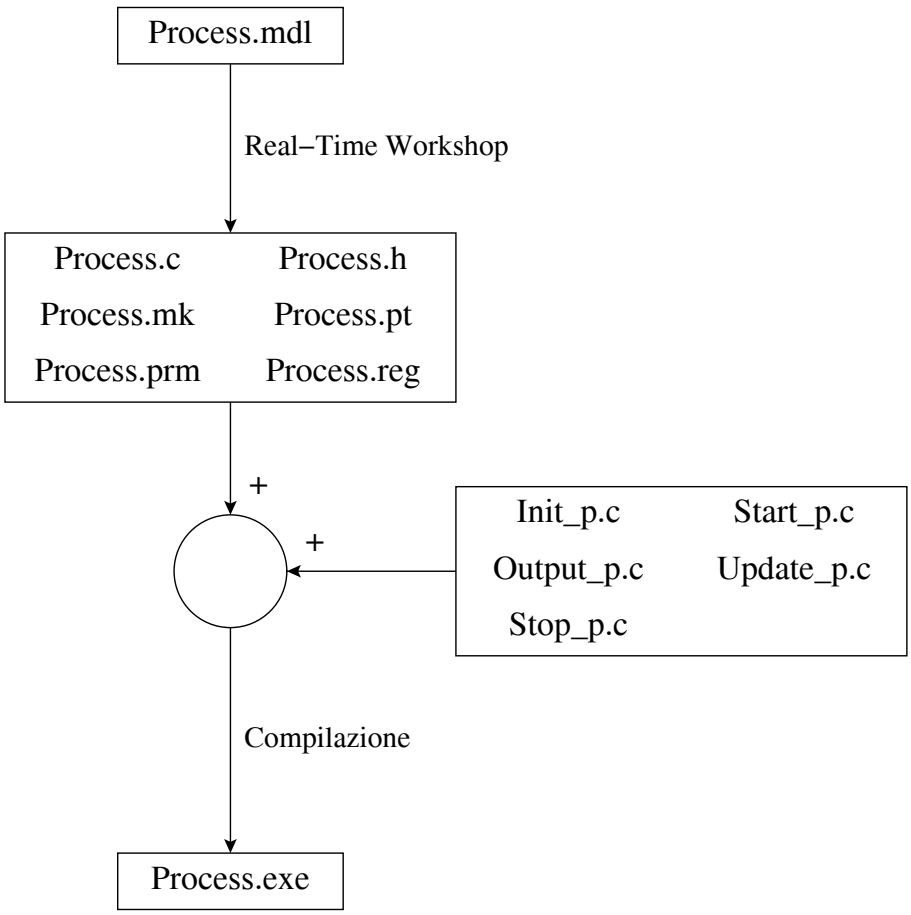

Figura 3.6: Fasi e files necessari alla creazione di Process.exe.

3.4.2 Funzionamento del programma

Una volta avviato Process, viene stabilita una connessione TCP con l'applet Telelab2 e vengono inviati tutti i dati relativi al riferimento ed ai parametri modificabili del controllore, dopodiché il server resta in attesa di ricevere o il segnale di avvio del processo o quello relativo ad una variazione delle caratteristiche sopra citate. Appena iniziato l'esperimento vengono trasmessi all'applet, ad intervalli di tempo prefissati, i dati relativi alla posizione, al riferimento ed al comando fornito dal controllore. Inoltre il programma resterà sempre in ascolto di eventuali segnali relativi alla modifica del riferimento o dei parametri variabili del controllore, che potranno subire variazioni in fase di esecuzione.
La terminazione del programma può verificarsi per varie cause, che possono essere la ricezione del segnale di fine da parte di Telelab2, la scadenza di un time-out od un errore generico dovuto per esempio ad una caduta della connessione. In tutti questi casi, una volta terminato Process, il controllo ritorna nuovamente al programma Interface.

3.5 Descrizione dell'applet Telelab2

L'applet Telelab2 è preposta al controllo del processo e viene avviata nel momento in cui termina Telalab1. Una volta stabilita la connessione con il programma Process, vengono ricevuti i dati riguardanti i vari riferimenti possibili, nonché i parametri modificabili del controllore. A questo punto viene visualizzata l'interfaccia utente (Fig. 3.7), che è suddivisa nelle seguenti sezioni:

- Gestione dei parametri modificabili del controllore.
- Gestione del riferimento.
- Visualizzazione grafica della dinamica dell'esperimento.
- Visualizzazione numerica della dinamica dell'esperimento.
- Pulsanti di inizio e fine esperimento.

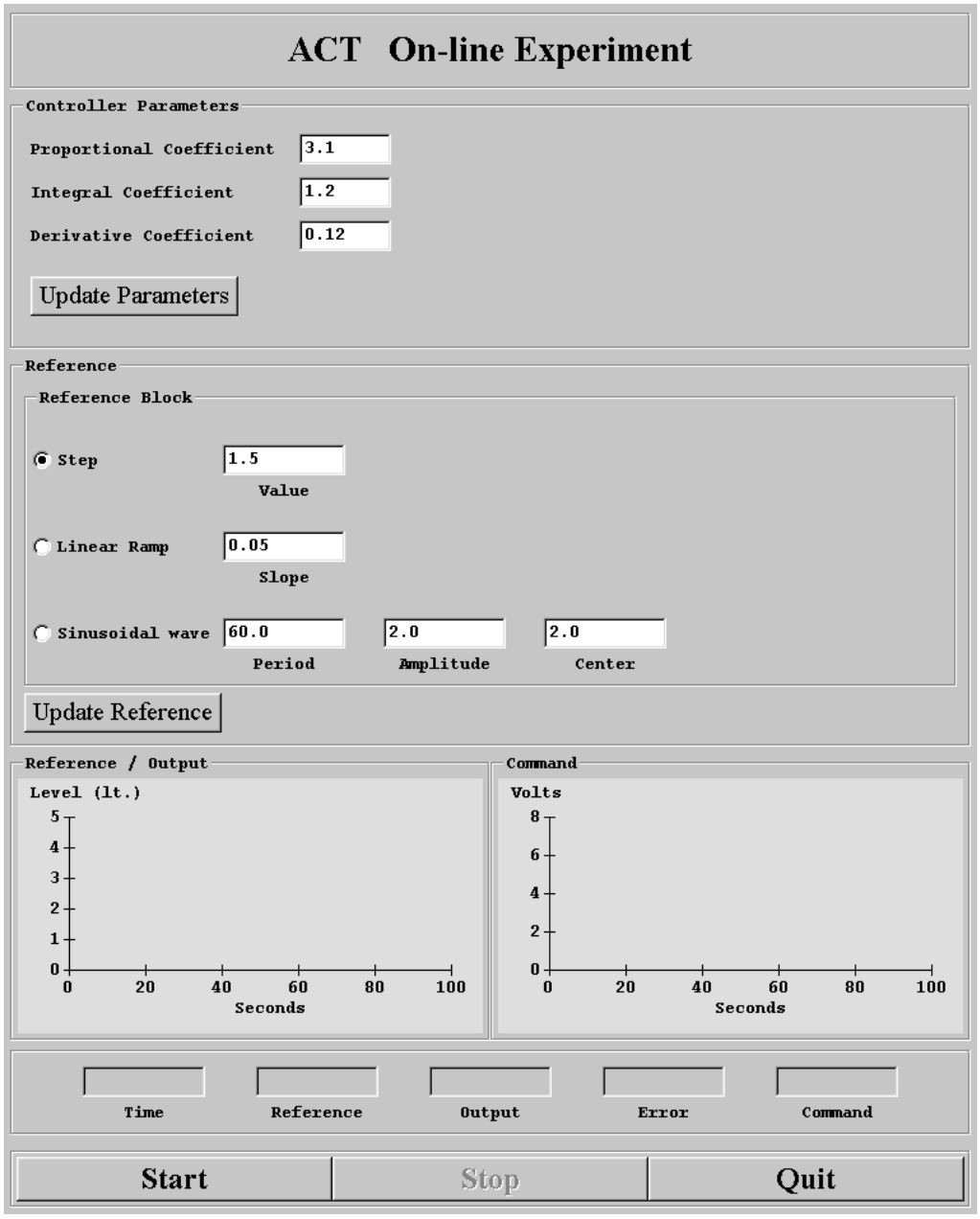

Figura 3.7: Interfaccia visuale dell'applet Telelab2.

È adesso compito dell'utente decidere quale operazione effettuare; ognuna di queste comporterà la trasmissione di un pacchetto TCP contenente il codice dell'operazione e gli altri dati necessari, dopodiché l'applet resterà in attesa della conferma dell'avvenuta ricezione da parte di Process.
Ogni eventuale anomalia dovuta ad un errore di processo o della connessione verrà segnalata mediante un'apposita finestra, dopodiché l'applet verrà terminata.
Nel caso invece che sia l'utente a fermare il processo, questa applet sarà terminata e verrà caricata una pagina in cui sarà possibile effettuare il download del file "Process.mat", contenente la dinamica dettagliata dell'esperimento in un formato utilizzabile da Matlab. Sarà così possibile per l'utente effettuare delle analisi fuori linea, ossia non in tempo reale, per poter ad esempio valutare con esattezza la validità e le caratteristiche del controllore.

Capitolo 4

Descrizione dettagliata dei moduli

In questo capitolo verranno trattati tutti i dettagli relativi all'implementazione del software, quali i codici di trasmissione, la gestione del tempo reale, le convenzioni adottate, ecc.

4.1 Comunicazione tra Interface e Telelab1

La prima connessione che viene stabilita è quella tra Interface e Telelab1, il cui scopo è quello di permettere la scelta del controllore da utilizzare.
Per convenzione le parole incluse in un pacchetto sono delimitate da un asterisco (*), mentre la fine del pacchetto è delimitata da un diesis (#); questo permetterà all'applicazione ricevente di scandire la stringa in arrivo e di distinguerne le varie componenti. La prima parola di un pacchetto sarà sempre il codice dell'operazione, che potrà essere seguito da altri parametri a seconda del tipo di operazione.
I codici relativi ai pacchetti trasmessi da Interface e Telelab1 sono riportati rispettivamente in tabella 4.1 e 4.2. La sequenza di utilizzo di tali codici varia a seconda che venga scelto un controllore predefinito oppure un controllore disegnato dall'utente.

CODE_TITLE	-T	Titolo del processo
CODE_CN	-N	Numero di controllori predefiniti
CODE_CONTR	-C	Nome controllore predefinito
CODE_FILE_READY	-D	Pronto a ricevere il controllore
CODE_USER_OK	-U	Esito della verifica utente (0=ok)
CODE_COMP_OK	-O	Esito della compilazione (0=ok)
CODE_PROC_OK	-K	Conferma esistenza del controllore (0=ok)
CODE_TIMEOUT	-I	Time-out scaduto

Tabella 4.1: Codici di trasmissione di Interface.

CODE_PERSONAL	-P	Dati personali utente
CODE_SC	-S	Codice controllore selezionato
CODE_FILE_SENT	-F	Richiesta di invio controllore
CODE_START	-A	Richiesta di avvio esperimento

Tabella 4.2: Codici di trasmissione di Telelab1.

Entrambi i casi sono stati raffigurati in figura 4.1 e 4.2, dove, per semplicità, si suppone che sia già stata effettuata la connessione TCP e che le fasi di compilazione e verifica utente e controllore abbiano avuto esito positivo. Per quanto riguarda l'interpretazione di queste figure è bene specificare che i pacchetti sono rappresentati dalle frecce, mentre il tempo scorre verso il basso lungo le linee verticali.

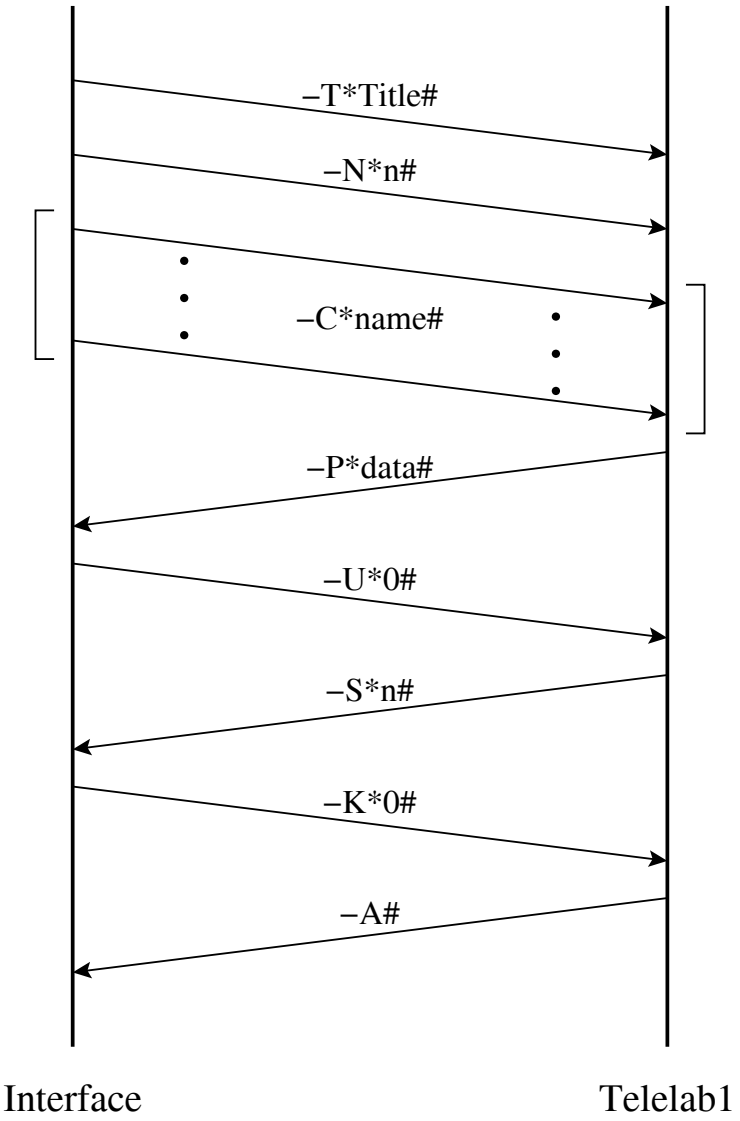

Figura 4.1: Scambio di messaggi nel caso di controllore predefinito.

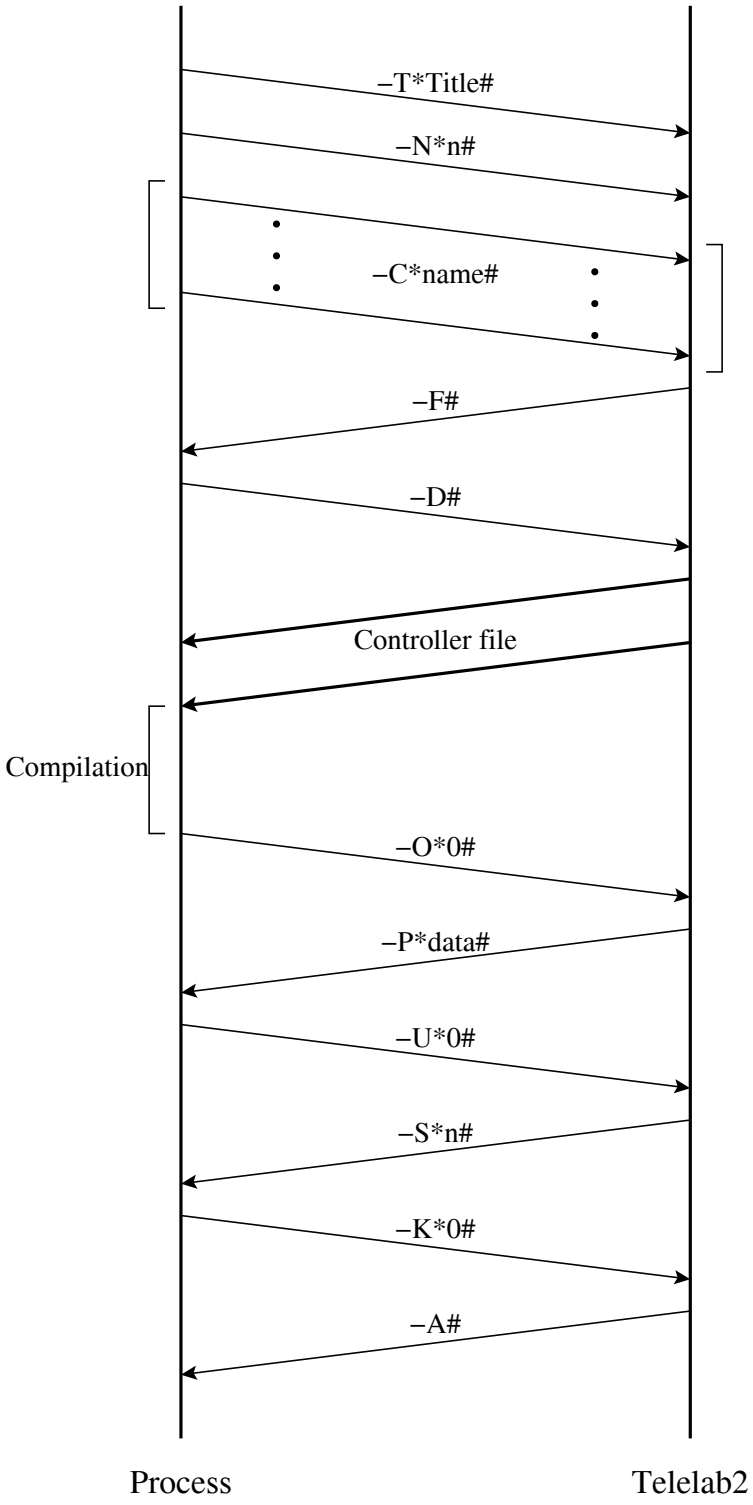

Figura 4.2: Scambio di messaggi nel caso di controllore disegnato dall'utente.

4.2 Comunicazione tra Process e Telelab2

La connessione TCP instaurata tra Process e Telelab2 è senz'altro più complessa della precedente, specialmente per quello che riguarda la fase di trasmissione da parte di Process dei dati relativi al riferimento.
Nelle seguenti sezioni saranno esposte in dettaglio tutte le peculiarità relative allo scambio di messaggi necessari a realizzare le varie operazioni.

4.2.1 Trasmissione dei parametri modificabili

Essendo Telelab2 un'applet destinata a gestire qualsiasi tipo di processo, per prima cosa resterà in attesa di ricevere tutte le informazioni inerenti il processo in esame. I primi dati che Process trasmetterà saranno quelli relativi ai parametri variabili del controllore; poiché ogni parametro consiste in uno scalare, associato solitamente ad una costante o ad un blocco di guadagno del modello Simulink, esso sarà reperibile tramite il corrispondente indice mediante un'opportuna istruzione fornita dal RTW. Infatti, tutte le costanti ed i blocchi di guadagno presenti in un modello vengono associati dal RTW ad un array (Tp_array) contenente, in ogni posizione, il nome del parametro, il valore, ed altri dati non rilevanti in questo contesto. Tra tutte le locazioni presenti nell'array verranno selezionate quelle relative ai parametri modificabili del controllore, riconoscibili in quanto il nome ad essi associato deve per convenzione iniziare con la stringa "ACT_TP_" (Automatic Control Telelab - Tuning Parameter); a questo punto tali dati verranno trasmessi all'applet mediante pacchetti TCP, il cui contenuto è raffigurato in figura 4.3.

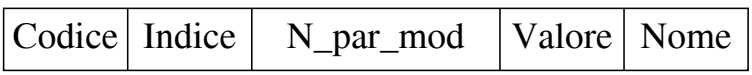

Figura 4.3: Struttura del pacchetto relativo alla trasmissione dei parametri modificabili del controllore.

I campi illustrati hanno il seguente significato:

Codice: codice dell'operazione (-1).

Indice: indice di Tp_array relativo al parametro in questione.

N_par_mod: numero totale di parametri modificabili.

Valore: valore del parametro.

Nome: nome del parametro.

L'elenco completo dei codici di trasmissione è contenuto in tabella 4.3 e verrà descritto più in dettaglio nelle prossime sezioni.

TUNING_PARAMETER_SIGNAL	-1	Parametri modificabili
REFERENCE_BLOCK_SIGNAL	-2	Blocco di riferimenti
REFERENCE_SIGNAL	-3	Singolo riferimento
REFERENCE_VALUE_SIGNAL	-4	Parametro interno al riferimento
STOP_INITIAL_SIGNAL	-5	Fine trasmissioni iniziali

Tabella 4.3: Codici di trasmissione dei dati iniziali di Process.

Una volta che i pacchetti sono giunti a destinazione sarà possibile da parte di Telelab2 visualizzare l'elenco dei parametri del controllore soggetti a modifica in fase di esecuzione (Fig. 4.4).

4.2.2 Convenzioni relative al riferimento

Terminata la fase precedente inizia quella relativa alla trasmissione dei riferimenti da parte di Process.
In generale sarà necessario un blocco di riferimento per ogni uscita presente nel sistema; all'interno di tale blocco saranno presenti i vari riferimenti applicabili, di cui solo uno alla volta sarà attivo. Quest'ultimo potrà consistere in uno o più numeri reali a seconda che si tratti di un gradino, una rampa, un'onda

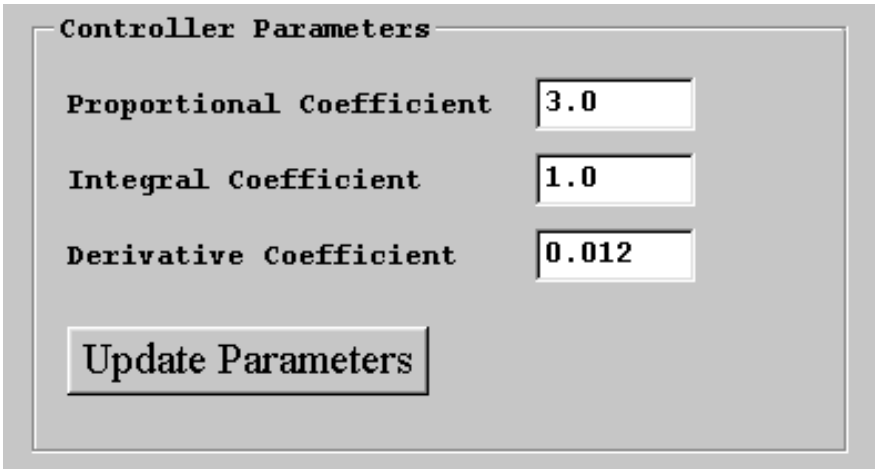

Figura 4.4: Parte dell'interfaccia grafica di Telelab2 inerente la gestione dei parametri modificabili del controllore.

sinusoidale od una funzione generica.
Esempi di ingressi applicabili sono rappresentati in figura 4.5 mentre in figura 4.6 è raffigurato l'interno del blocco Simulink che permette di poter scegliere, sulla base del valore di una costante, uno dei tre tipi di riferimento standard (gradino, rampa lineare, sinusoide). Per poter permettere al programma di individuare quali blocchi riguardano il riferimento, è necessario adottare delle convenzioni, che consistono nel far precedere l'etichetta relativa ad un blocco da una determinata stringa; la conoscenza di queste convenzioni non riguarda l'utente finale, bensì colui che avrà il compito di gestire il tele-laboratorio. Di seguito sono riportati tutti i prefissi relativi ai riferimenti:

ACT_RS_: Questo prefisso indica che il blocco in questione è il selettore di uno switch di riferimenti (Reference Selector), e potrà assumere i valori interi compresi tra 1 ed il numero di ingressi totali. La sua etichetta coinciderà con il nome associato al blocco di riferimento, mentre il suo valore identificherà il riferimento attivo.

ACT_NO_: Dal momento in cui tutte le costanti ed i blocchi di guadagno interni ad un riferimento vengono automaticamente inseriti nell'interfaccia visuale dell'applet, può rendersi necessario il caso di non volerne

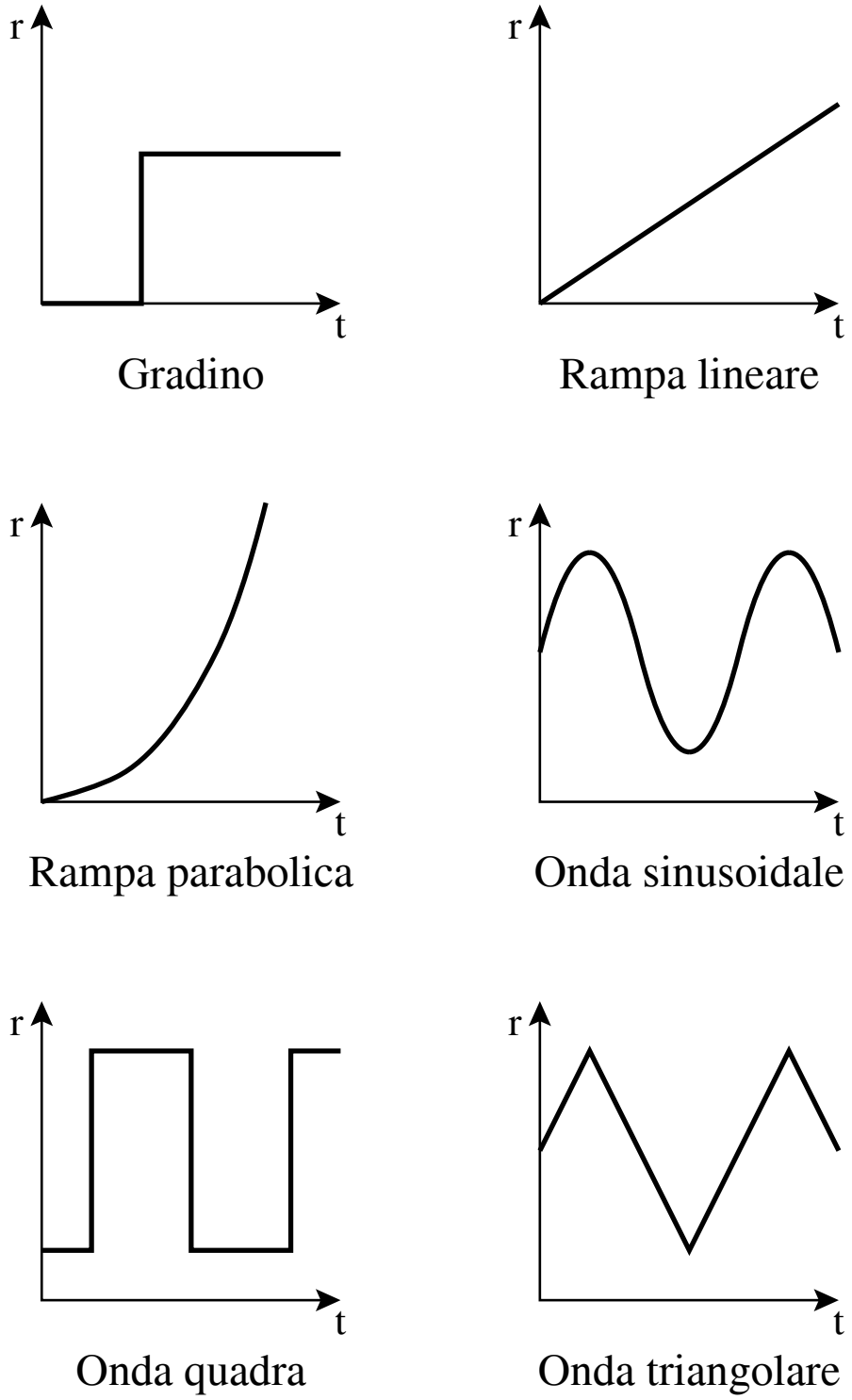

Figura 4.5: Esempi di possibili funzioni di ingresso.

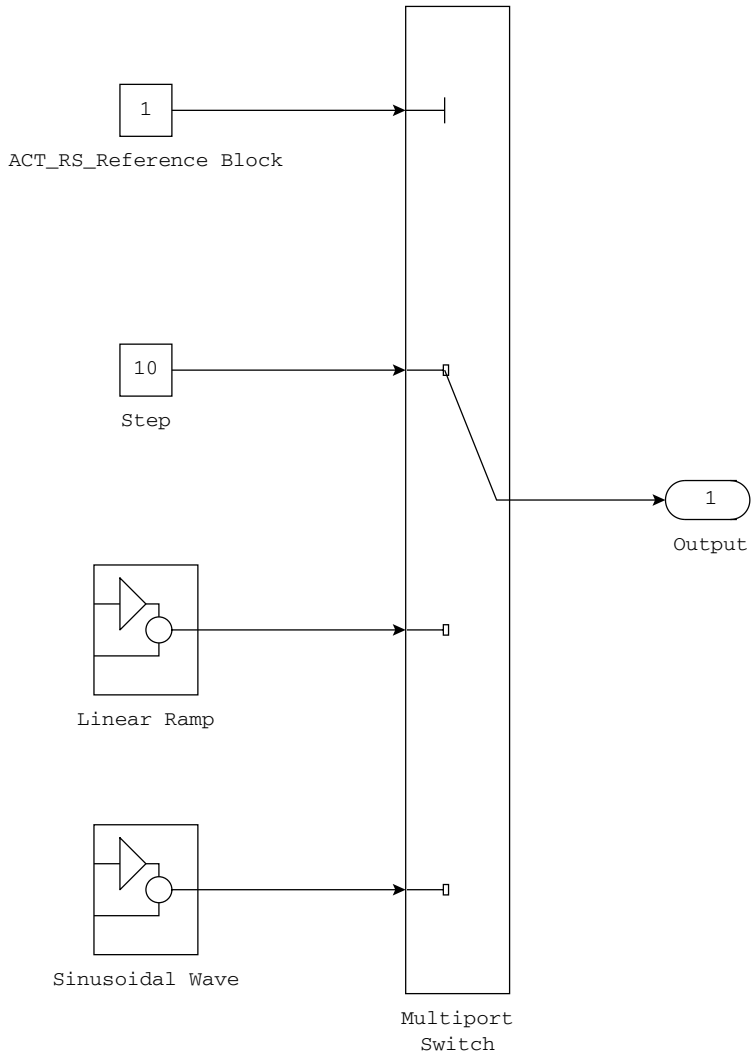

Figura 4.6: Interno del blocco Simulink preposto alla scelta del riferimento.

visualizzare qualcuno, in quanto usato come blocco il cui valore non deve essere cambiato; in questo caso basterà inserire questo prefisso affinché l'etichetta non compaia nell'interfaccia.

ACT_NEVER_: Questo prefisso funziona allo stesso modo di quello appena citato, con la sola differenza che, mentre il precedente agisce su blocchi elementari, questo opera su strutture più complesse; ad esempio se questo prefisso viene impiegato in un blocco che ne include altri (subsystem), nessuno di questi ultimi comparirà nell'interfaccia visuale.

ACT_NEVER_TIME: Questa etichetta, a differenza delle precedenti, non può essere utilizzata come prefisso, bensì deve comparire esattamente così com'è. Il blocco costante ad essa associato corrisponde al tempo di simulazione in cui l'utente cambia il tipo di riferimento. Questo può essere utile per realizzare delle strutture che altrimenti risulterebbero di difficile se non impossibile implementazione. Ad esempio il blocco che realizza una rampa lineare (Fig. 4.7) ha la necessità di conoscere il tempo ed il valore del riferimento al momento in cui l'utente decide di inviare tale ingresso (l'unico parametro modificabile sarà la pendenza), che potranno quindi essere acquisiti mediante l'utilizzo di questa etichetta e della successiva. Questo in quanto la rampa deve cominciare esattamente dal punto in cui si trovava il riferimento al momento della variazione.

ACT_NEVER_VALUE: Questa etichetta agisce allo stesso modo della precedente, con la sola differenza che contiene il valore del riferimento nel momento in cui questo è stato variato.

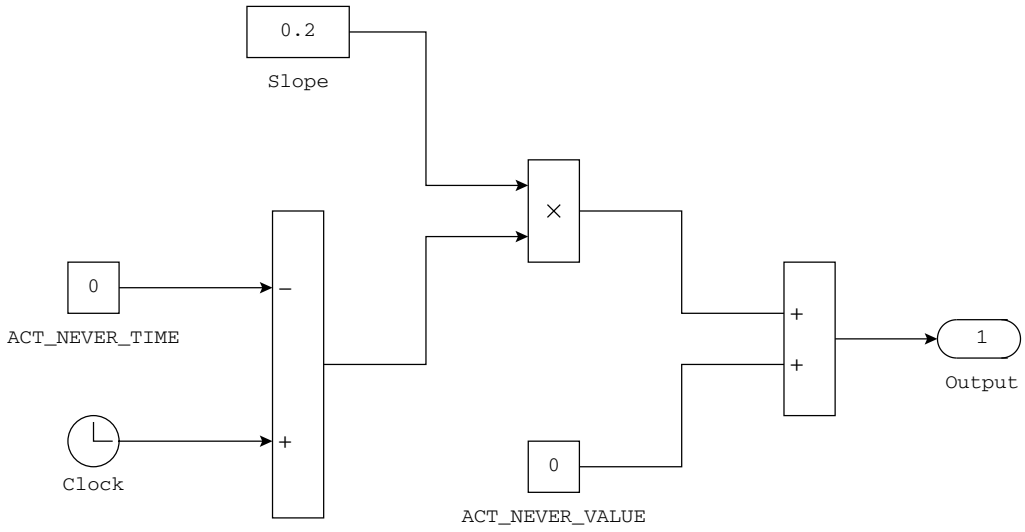

Figura 4.7: Interno del blocco relativo alla generazione di una rampa lineare.

4.2.3 Trasmissione del riferimento

Una volta trasmessi i dati relativi ai parametri modificabili del controllore, Process scandisce il modello Simulink alla ricerca dei blocchi riguardanti i riferimenti, dopodiché, previa una fase di elaborazione, procede alla trasmissione dei dati verso l'applet Telelab2. I codici di operazione sono riportati in tabella 4.3, mentre di seguito viene esposta la descrizione dei campi dei pacchetti utilizzati.
Per ogni blocco di riferimento vengono spediti i seguenti pacchetti:

- pacchetto relativo al blocco di riferimento (Fig. 4.8), i cui campi hanno il seguente significato:

  Codice: codice dell'operazione (-2).

  Num_blocco: numero progressivo indicante il blocco contenente i riferimenti.

  Indice_RS: indice di Tp_array relativo al selettore dei riferimenti.

  Valore_RS: valore del selettore.

  N_rif_tot: numero totale dei riferimenti contenuti nel blocco.

  Nome: nome del blocco dei riferimenti.

Codice	Num_blocco	Indice_RS	Valore_RS	N_rif_tot	Nome

Figura 4.8: Struttura del pacchetto relativo al blocco di riferimento.

- pacchetto relativo al singolo riferimento selezionabile (Fig. 4.9), i cui campi sono i seguenti:

  Codice: codice dell'operazione (-3).

  Blocco_rif: numero del blocco a cui appartiene il riferimento.

  Indice_rif: indice relativo a questo riferimento.

  Nome_rif: nome del riferimento.

Codice	Blocco_rif	Indice_rif	Nome_rif

Figura 4.9: Struttura del pacchetto relativo al singolo riferimento.

- pacchetto relativo ad un campo interno ad un singolo riferimento selezionabile (Fig. 4.10), come per esempio il campo "frequenza" che è interno al riferimento "Onda Sinusoidale".

  Codice: codice dell'operazione (-4).

  Blocco_rif: numero del blocco a cui appartiene il riferimento.

  Indice_rif: indice relativo al riferimento a cui appartiene il campo.

  Indice: indice di Tp_array relativo a questo campo.

  Valore: valore del campo.

  Nome: nome del campo.

Codice	Blocco_rif	Indice_rif	Indice	Valore	Nome

Figura 4.10: Struttura del pacchetto relativo ad un campo interno ad un singolo riferimento.

Al termine della trasmissione del riferimento viene inviato un pacchetto con codice "STOP_INITIAL_SIGNAL", che avvisa Telelab2 del termine della fase iniziale di comunicazione; a questo punto sarà possibile visualizzare i possibili riferimenti applicabili al processo (Fig. 4.11), che saranno poi inseriti in un'opportuna struttura dati in modo da poterne consentire un'agevole modifica in fase di esecuzione.

4.2.4 Modifica dei parametri del controllore e del riferimento

Appena completata la fase di trasmissione dei dati iniziali precedentemente descritta, l'applet Telelab2 è pronta ad avviare l'esperimento, così come a

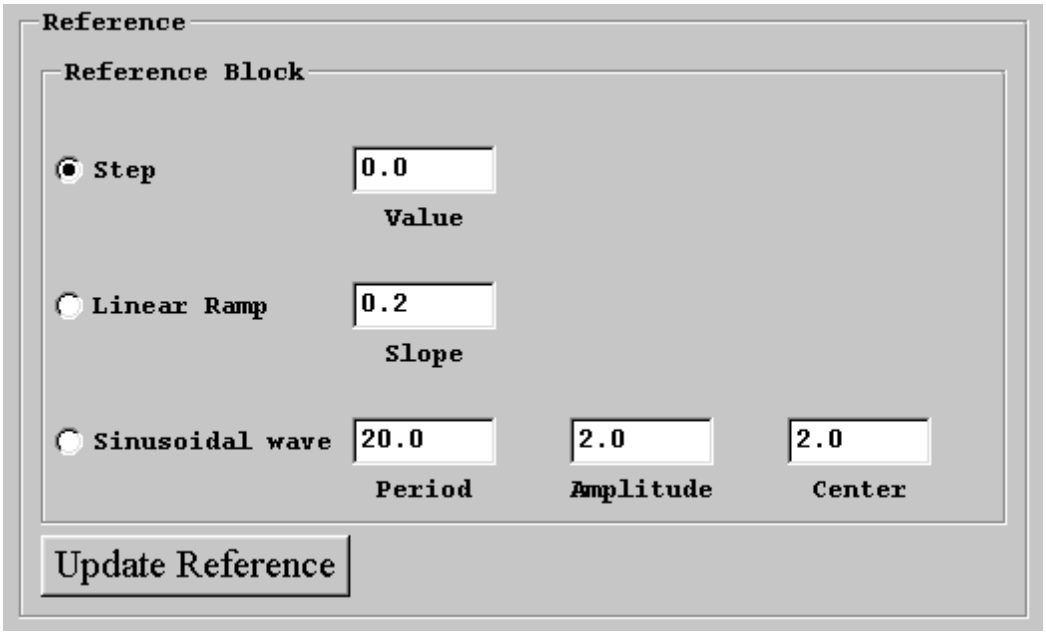

Figura 4.11: Parte dell'interfaccia grafica di Telelab2 inerente la gestione del riferimento.

modificarne i parametri.
Per quanto riguarda l'inizio e la fine dell'esperimento verranno semplicemente trasmessi due pacchetti contenenti solamente il relativo codice dell'operazione (Tab. 4.4).
Nel caso vengano modificati uno o più parametri del controllore sarà trasmesso un pacchetto con il formato dei campi come in figura 4.12, il cui significato è il seguente:

n: numero di parametri modificabili totali.

Indice_i: indice di Tp_array relativo all'i-esimo parametro.

Valore_i: valore del parametro.

Poiché gli indici di cui sopra sono certamente positivi (od al più nulli), non sarà possibile un'ambiguità con i codici di operazione descritti in tabella 4.3 e 4.4, in quanto trattasi di numeri negativi o caratteri alfanumerici.

START_SIGNAL	-10	Richiesta di inizio esperimento
STOP_SIGNAL	-20	Richiesta di fine esperimento
DATA_RECEIVED_SIGNAL	r	Pacchetto ricevuto
OUTPUT_SIGNAL	o	Invio dinamica del processo
END_SIGNAL	e	Esperimento terminato
TIMEOUT_SIGNAL	t	Time-out scaduto

Tabella 4.4: Altri codici di trasmissione.

Indice_1	Valore_1	. . .	Indice_n	Valore_n

Figura 4.12: Struttura del pacchetto relativo alla modifica dei parametri del controllore.

La stessa procedura si ripete in modo equivalente per quanto concerne la modifica del riferimento; in questo caso la lunghezza del pacchetto dipenderà dal numero di campi presenti nel nuovo riferimento a cui bisogna aggiungere quelli relativi al blocco selettore. Un gradino, ad esempio, conterrà due sole coppie indice-valore, mentre un'onda sinusoidale ne avrà quattro (selettore, centro, ampiezza, frequenza).
Una volta che questi pacchetti giungono a Process, l'operazione di elaborazione sarà minima, poiché sono già presenti in modo esplicito sia le posizioni dell'array che devono essere aggiornate, sia i nuovi valori che dovranno assumere. Tutto questo al fine di non occupare preziose risorse di calcolo che dovranno essere dedicate al controllo del processo in tempo reale.
Per notificare la corretta ricezione del pacchetto, Process provvederà ad inviarne uno a Telelab2 con codice DATA_RECEIVED_SIGNAL. Fino a quando Telelab2 non avrà ricevuto tale pacchetto di conferma non sarà possibile modificare alcun parametro né alcun riferimento.

4.2.5 Trasmissione delle uscite

Oltre al controllo del processo, il programma Process trasmetterà, ad intervalli di tempo costanti, i dati relativi all'esperimento in corso, ossia l'uscita, il riferimento ed il comando. Il formato del pacchetto utilizzato in questa fase è rappresentato in figura 4.13, ed è relativo ad un processo di tipo SISO (Simple Input, Simple Output). I campi hanno il seguente significato:

Codice: codice dell'operazione (o).

Tempo: istante di tempo a cui si riferiscono i dati.

Riferimento: riferimento applicato in quell'istante di tempo.

Uscita: uscita del processo.

Comando: uscita del controllore.

Codice	Tempo	Riferimento	Uscita	Comando

Figura 4.13: Struttura del pacchetto riguardante la trasmissione dei dati relativi all'esperimento.

Una volta che Telelab2 avrà ricevuto questo pacchetto, provvederà a visualizzarne i dati, nonché ad aggiornare i grafici in cui viene rappresentata la dinamica dell'esperimento in corso (Fig. 4.14).

4.3 Gestione del tempo reale

Un aspetto fondamentale del tele-laboratorio è quello di riuscire a controllare il processo in tempo reale, ossia di far coincidere il tempo utilizzato per effettuare i calcoli del controllore (tempo simulato) con quello reale. Un'eventuale perdita di tale proprietà comporterebbe sicuramente dei ritardi sia nell'acquisizione delle uscite che nell'azionamento degli attuatori, con una possibile perdita di stabilità del sistema controllato.

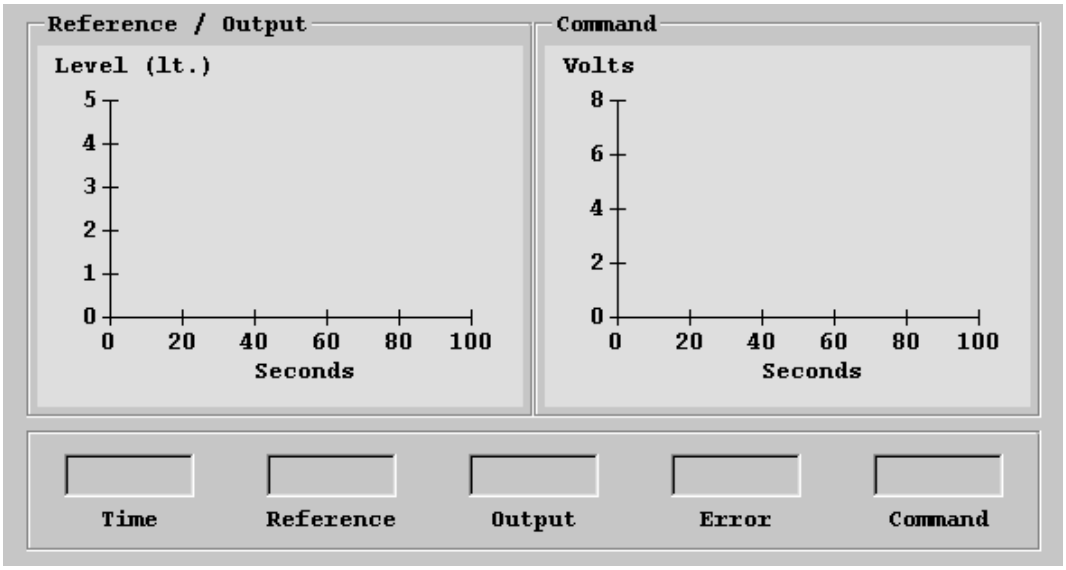

Figura 4.14: Valori e grafici rappresentanti l'andamento dell'esperimento in tempo reale.

Poiché il Real-Time Workshop converte un modello Simulink in un file eseguibile che non gestisce il tempo reale, si è reso necessario inserire delle opportune linee di codice C all'interno dei file sorgenti ottenuti con il RTW.
Il sistema operativo utilizzato per la gestione del tele-laboratorio è Windows 95/98 della Microsoft; tale sistema operativo non contiene istruzioni sofisticate preposte alla temporizzazione dei processi in esecuzione, per cui non è stato possibile ottenere il real-time mediante chiamate a routine di sistema. I problemi da risolvere in questa fase sono i seguenti:

1. far coincidere il tempo simulato con quello reale (temporizzazione);

2. impedire al sistema operativo o ad altre applicazioni presenti di utilizzare troppe risorse di calcolo, rischiando di compromettere il punto precedente;

3. poter ridurre il più possibile il tempo di campionamento senza invalidare i due punti sopra citati.

Per quanto riguarda i problemi descritti nei punti 2 e 3, essi sono stati risolti aumentando al massimo la priorità di Process rispetto agli altri programmi in

esecuzione; in questo caso il sistema operativo dedicherà la quasi totalità del tempo macchina a Process, il quale ne potrà disporre per effettuare i calcoli relativi al controllo del processo reale; inoltre saranno disattivate tutte le altre funzioni non vitali del sistema operativo.
A questo punto resta il problema della temporizzazione, ossia di eseguire certe istruzioni ad istanti di tempo prefissati; ad esempio, se il tempo di campionamento del processo è di 100 millisecondi, è necessario che ogni acquisizione delle uscite del sistema disti dalla precedente esattamente di 100 millisecondi. Per ottenere questo risultato è stato necessario aggiungere un ciclo di "attesa attiva", ossia un ciclo a vuoto che termina solamente al verificarsi di un certo evento, che in questo caso coincide con l'uguaglianza tra il tempo simulato e quello reale.
Il diagramma di flusso che riassume le principali operazioni compiute da Process relativamente alla gestione del tempo reale è descritto in figura 4.15.
Il frammento di codice preposto a questo compito è il seguente:

```
target_clock = (ssGetT(s) * rps) + initial_clock;
if (target_clock < ms_clock())
    loop_error += (ms_clock() - target_clock)
else
    while (target_clock > ms_clock()) ;
```

La prima linea calcola l'istante di tempo in cui il programma dovrà proseguire (target_clock) affinché si mantenga una coerenza tra il tempo simulato (ssGetT(s)) e quello reale. L'istruzione di confronto controlla se il tempo reale (ms_clock()) è maggiore di quello simulato; se ciò si verifica significa che ci troviamo di fronte ad una momentanea perdita del real-time, per cui viene incrementato un opportuno contatore di errore (loop_error). Se tale contatore raggiunge una determinata soglia l'esperimento viene terminato. Nel caso invece che il confronto dia esito negativo, verrà eseguito il ciclo di "attesa attiva" al termine del quale il programma riprenderà il suo svolgimento.

Se si verifica la condizione di loop_error significa che in un istante di campionamento il calcolatore non riesce ad eseguire tutti i calcoli necessari per l'acquisizione dei dati ed il controllo del processo; un possibile rimedio può essere quello di aumentare il tempo di campionamento, purché questo non pregiudichi il controllo del sistema. In caso contrario, significa che il calcolatore non dispone di sufficiente potenza per poter eseguire certe istruzioni in un determinato lasso di tempo, per cui la soluzione può essere quella di sostituirlo con un altro di maggiore potenza, o di semplificare la legge di controllo, se possibile.

Come accennato in precedenza il sistema operativo Windows 95/98 non è stato progettato per garantire una temporizzazione tra i processi in esecuzione. Il problema in questa tesi è stato risolto mediante l'ausilio di un ciclo di attesa attiva ed impostando un'alta priorità a Process. Un'altra possibile soluzione sarebbe quella di adottare un diverso sistema operativo, oppure un opportuno kernel che garantisca il tempo reale. Tra questi possiamo citare "Tornado", un ambiente integrato che include VxWorks (un sistema operativo ad elevate prestazioni real-time), un insieme di tools per la costruzione di applicazioni (compilatori, linkers e utilità di archiviazione) ed un ambiente di sviluppo interattivo (editor, debugger, strumenti di configurazione e browser), le cui caratteristiche risultano essere molto somiglianti a quelle del sistema operativo Unix. In questo caso saranno messe a disposizione delle opportune istruzioni che si preoccuperanno di risolvere tutti i problemi legati al tempo reale.

Esiste inoltre un recente toolbox di Matlab chiamato "Real-Time Windows Target" che consente l'esecuzione di un modello Simulink in tempo reale; il problema di tale toolbox è che non può essere convertito in un eseguibile e che necessita per il suo funzionamento dell'interazione in locale con l'utente, per cui non è possibile integrarlo con le altre routine di gestione del tele-laboratorio, come ad esempio quelle relative alla connessione TCP.

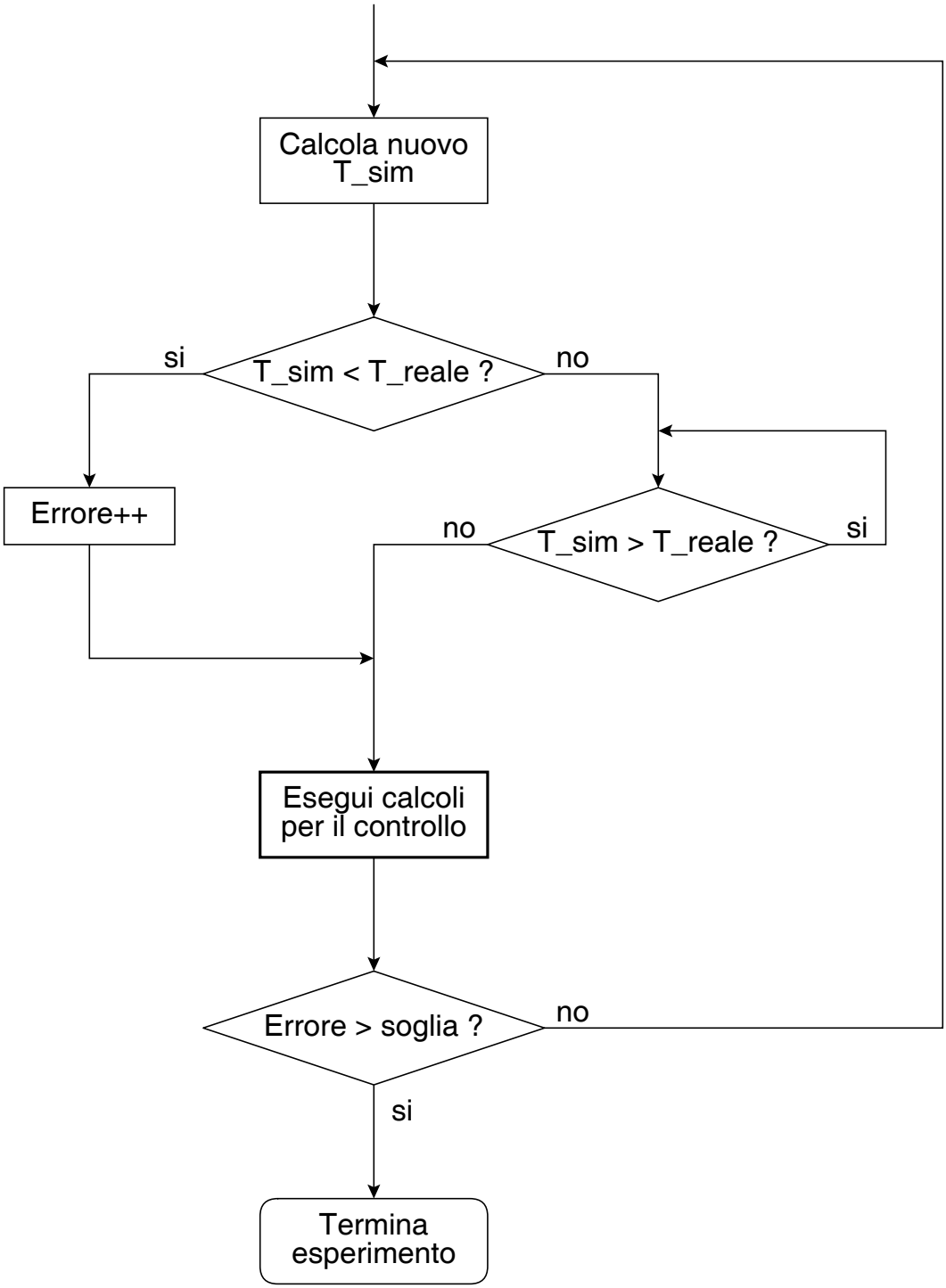

Figura 4.15: Diagramma di flusso semplificato relativo alla gestione del tempo reale da parte di Process.

Capitolo 5

Descrizione di un esperimento

In questo capitolo verrà descritto nel dettaglio il funzionamento di un processo collegato al tele-laboratorio. Saranno trattati gli aspetti relativi alla modellizzazione del sistema ed alla progettazione del controllore. Al termine verranno esposti i risultati sperimentali ottenuti.

5.1 Descrizione fisica del processo

L'esperimento che sarà esposto in questo capitolo è quello relativo ad un controllo di livello di un fluido all'interno di un determinato contenitore [7]; in questo caso il fluido adottato è acqua. Oltre al recipiente principale ne esiste uno secondario con lo scopo di contenere l'acqua necessaria per il funzionamento. L'apparecchiatura in questione è schematizzata in figura 5.1, dove è possibile notare la vasca superiore opportunamente graduata (a), ed il serbatoio di recupero inferiore (b). Il livello raggiunto dall'acqua viene misurato mediante un trasduttore di pressione posto alla base della vasca superiore (c) che, mediante un opportuno condizionatore di segnale, genererà una tensione proporzionale alla pressione esercitata dalla colonna d'acqua. L'organo di attuazione del processo è costituito da una pompa ad immersione (d) inserita all'interno del serbatoio di recupero, che provvederà a far risalire l'acqua fino alla sommità della vasca superiore.
Sia il trasduttore di pressione che la pompa sono stati collegati ad una scheda di

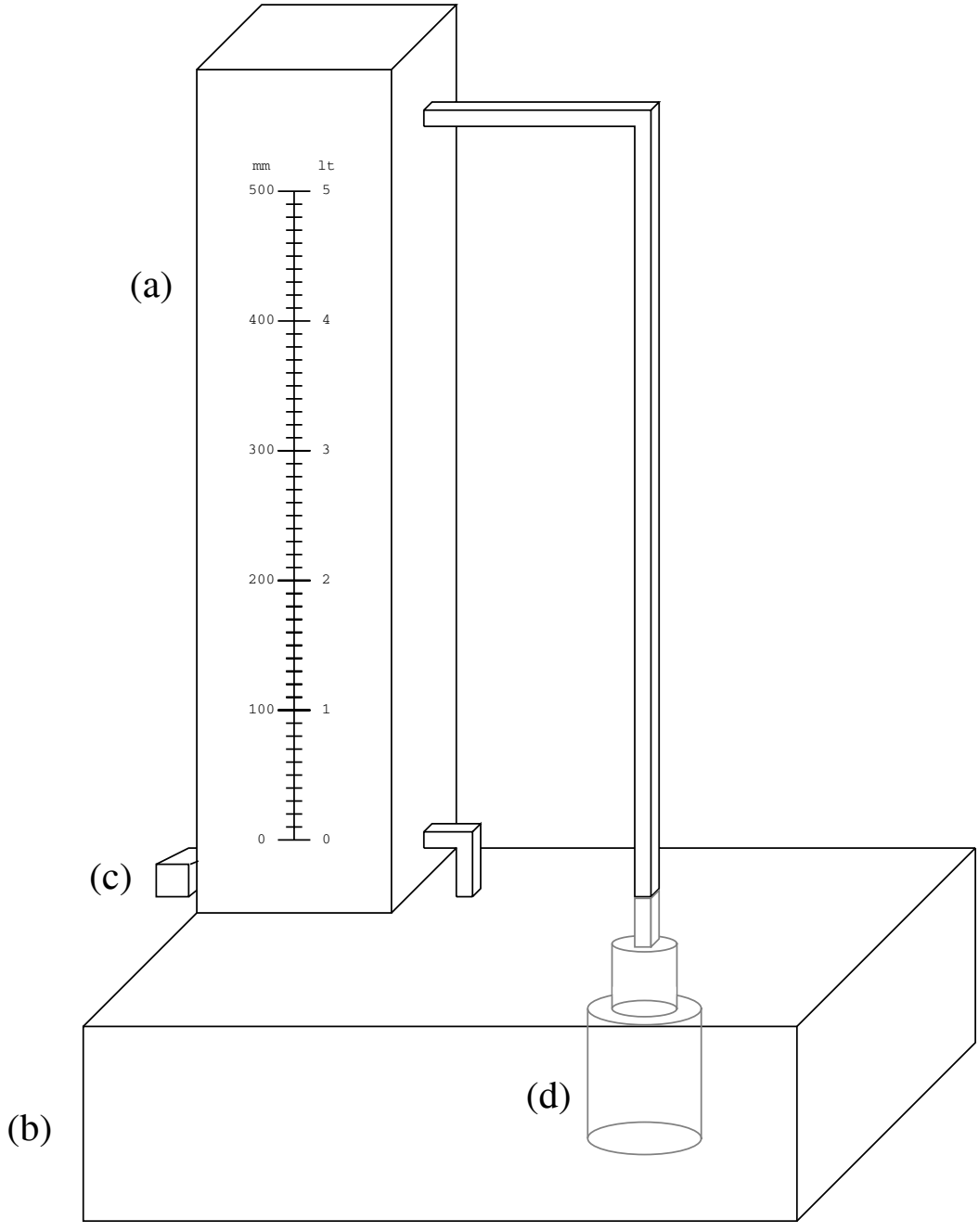

Figura 5.1: Processo fisico per il controllo di livello.

acquisizione dati, rispettivamente all'ingresso analogico ed all'uscita analogica, in modo da poter elaborare tali segnali tramite un calcolatore.

5.2 Costruzione del modello matematico

Per progettare ed analizzare il sistema di controllo, è necessario costruire un modello matematico del sistema che rappresenti in modo sufficientemente fedele il processo fisico.
Nella tabella 5.1 sono descritti i dati tecnici relativi alle dimensioni delle vasche ed alle tensioni applicabili.

Massima capacità della vasca	5 lt
Sezione della vasca	0.01 m^2
Raggio interno del tubo di scarico	2.4 mm
Massima tensione applicabile alla pompa	8 V

Tabella 5.1: Dati tecnici inerenti il processo fisico.

5.2.1 Studio della dinamica libera del sistema

Come prima cosa possiamo analizzare il sistema nella sua evoluzione libera, ossia in assenza di organi di attuazione. Verrà fatto riferimento alla tabella 5.2 per quanto riguarda i nomi delle variabili adottate.

h	Livello del liquido
A_1	Sezione della vasca
A_2	Sezione del tubo di scarico
P_a	Pressione atmosferica
ρ	Densità del liquido

Tabella 5.2: Variabili utilizzate per lo studio del modello.

Per l'equazione di continuità si ha:

$$A_1\, v_1 = A_2\, v_2$$

ossia

$$v_2 = \frac{A_1}{A_2}\, v_1$$

dove v_1 e v_2 sono rispettivamente le velocità del liquido nella vasca superiore e nel tubo di scarico.
Scrivendo l'equazione di Bernoulli per i due punti sopra citati otteniamo:

$$P_a + \rho\, g\, h + \frac{1}{2}\,\rho\, v_1^2 = P_a + \frac{1}{2}\,\rho\, v_2^2$$

semplificando si ricava:

$$2\, g\, h + v_1^2 = v_2^2$$

Sostituendo il valore di v_2 calcolato in precedenza si ha:

$$2\, g\, h + v_1^2 = \left(\frac{A_1}{A_2}\right)^2 v_1^2$$

ossia:

$$v_1 = \sqrt{\frac{2\, g\, h}{\left(\frac{A_1}{A_2}\right)^2 - 1}} = k\,\sqrt{h}$$

con

$$k = \sqrt{\frac{2\, g}{\left(\frac{A_1}{A_2}\right)^2 - 1}}$$

Poiché la velocità non è altro che la derivata dello spazio rispetto al tempo possiamo scrivere:

$$v_1 = -\frac{dh}{dt} = -\dot{h}$$

$$\boxed{\dot{h} = -k\,\sqrt{h}}$$

Questa risulta quindi essere l'equazione differenziale rappresentante la dinamica libera del sistema; tale equazione è non lineare vista la presenza dell'operatore di radice quadrata.

5.2.2 Studio della dinamica forzata del sistema

A questo punto si rende necessaria l'analisi della dinamica del sistema in presenza di attuazione, ossia della pompa. Da misure sperimentali è risultato che tale componente non risponde in modo lineare alla tensione che gli viene applicata. È infatti presente, oltre ad una soglia, anche un comportamento non perfettamente lineare della portata di acqua in funzione della tensione in ingresso.
Le misure sperimentali della portata sono illustrate nella tabella 5.3 e rappresentate graficamente in figura 5.2.

Tensione (V)	Portata (m^3/s)
≤ 3.7	0
5	$2.44 \cdot 10^{-5}$
6	$3.50 \cdot 10^{-5}$
7	$4.65 \cdot 10^{-5}$
8	$5.55 \cdot 10^{-5}$

Tabella 5.3: Portata della pompa per varie tensioni.

La portata di acqua in funzione della tensione può essere quindi approssimata tramite la seguente equazione in cui compare l'effetto dovuto alla soglia, ma è stata tolta l'altra non-linearità tramite l'approssimazione con una retta interpolante (figura 5.3).

$$q = \begin{cases} 0 & \text{se } V \leq 3.7 \\ -5.032 \cdot 10^{-5} + 1.36 \cdot 10^{-5} V & \text{se } V > 3.7 \end{cases}$$

Per rendere omogenee le dimensioni di una portata con quelle di una velocità è necessario dividerle per una sezione; il modello completo del sistema risulterà quindi essere:

$$\boxed{\dot{h} = -k\sqrt{h} + q/A_1}$$

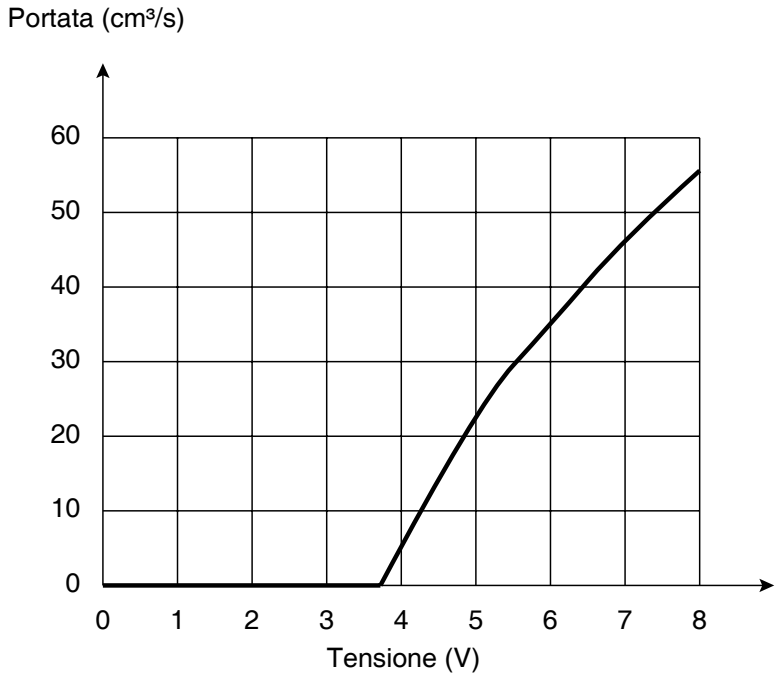

Figura 5.2: Caratteristica dell'attuatore (pompa).

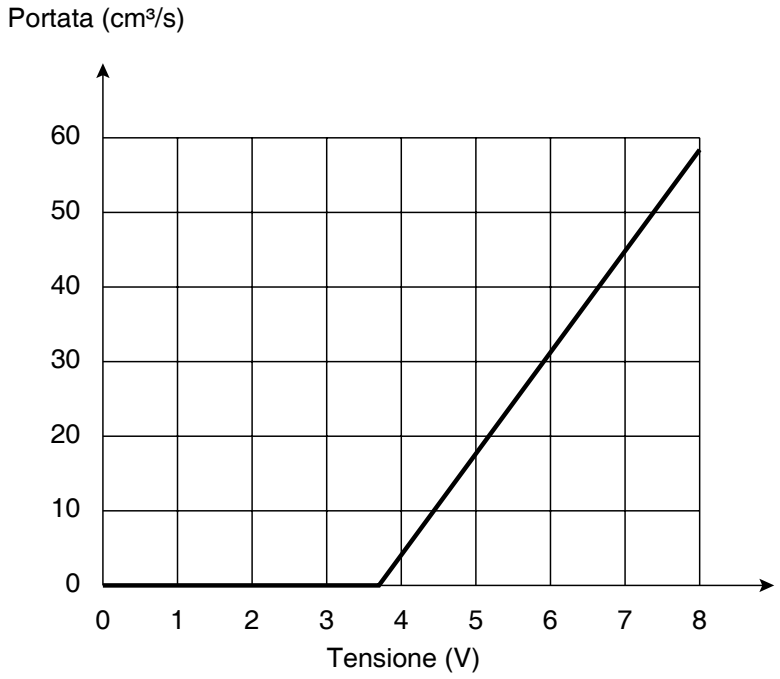

Figura 5.3: Caratteristica approssimata dell'attuatore (pompa).

5.3 Sintesi del controllore

Una volta determinato il modello matematico rappresentante il processo, è possibile dedicarsi alla ricerca di una legge di controllo che soddisfi certi requisiti. Tali requisiti possono essere il tempo di salita, la massima sovraelongazione, il tempo di assestamento, ecc.
I sistemi di controllo più evoluti sono quelli a catena chiusa (o retroazionati), in cui l'azione del controllo dipende in qualche modo dall'uscita (oltre che dal riferimento); i vantaggi fondamentali di questi controllori rispetto a quelli a catena aperta sono fondamentalmente due:

- Minore sensibilità alle variazioni parametriche.
- Minori effetti di grandezze disturbanti.

L'importanza di questi due vantaggi risulta ulteriormente chiarita dal fatto che variazioni parametriche e disturbi sono in generale dovuti ad eventi di carattere aleatorio e quindi imprevedibili se non nelle loro caratteristiche statistiche. In generale un controllo in retroazione può essere schematizzato come in figura 5.4, in cui il controllore dovrà fornire un opportuno comando al fine di ridurre o azzerare l'errore.

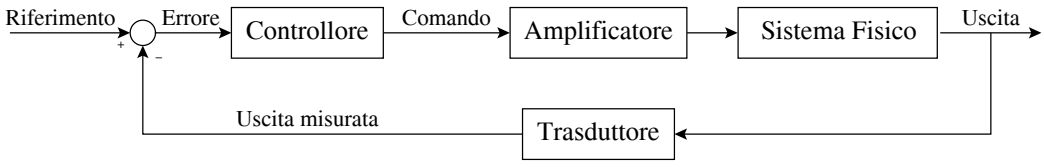

Figura 5.4: Schema di un controllo in retroazione.

Vista la varietà di regolatori che possono essere progettati per il controllo di un processo, verrà descritta in dettaglio solamente una legge di controllo, quella relativa ai controllori di tipo P.I.D. (Proporzionali - Integrali - Derivativi), che risultano essere quelli normalmente impiegati nell'industria.
In seguito verranno confrontati i valori ottenuti da tale controllore con quelli relativi ad una tecnica di controllo più evoluta, basata sulla linearizzazione in retroazione.

5.3.1 Caratteristiche di un controllore P.I.D.

Come enunciato in precedenza un controllore di tipo P.I.D. è il risultato dell'unione delle seguenti tre componenti [1]:

Azione Proporzionale (P): è l'azione introdotta da un amplificatore o da un attenuatore; l'andamento nel tempo della variabile di controllo è dato da:

$$u(t) = K_p\, e(t)$$

la funzione di trasferimento di tale blocco è:

$$W(s) = K_p$$

L'uscita, a parte il coefficiente moltiplicativo, è una copia perfetta dell'ingresso.

Azione Integrale (I): questa azione viene introdotta da un integratore puro, il cui andamento è:

$$u(t) = K_i \int_{t_0}^{t} e(\tau)\, d\tau$$

la cui funzione di trasferimento vale:

$$W(s) = \frac{K_i}{s}$$

Il termine K_i è denominato "coefficiente dell'azione integrale".

Azione Derivativa (D): è l'azione introdotta da un derivatore puro; la variabile di controllo ha il seguente andamento:

$$u(t) = K_d\, \frac{de(t)}{dt}$$

La funzione di trasferimento risulta essere:

$$W(s) = s \cdot K_d$$

dove K_d è il "coefficiente dell'azione derivativa".

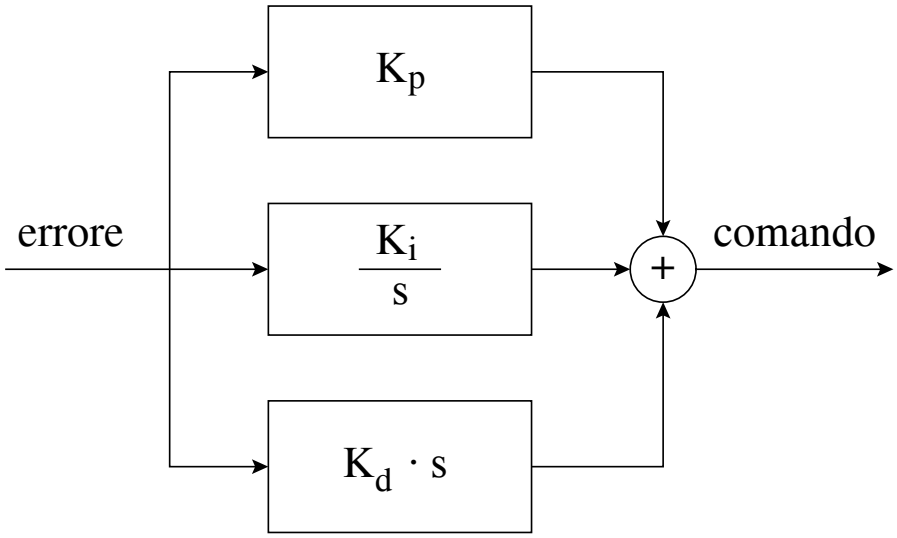

Figura 5.5: Schema a blocchi di un controllore P.I.D.

Lo schema completo di un controllore P.I.D. è riassunto in figura 5.5.
La funzione di trasferimento totale del controllore risulta essere:

$$W(s) = K_p + \frac{K_i}{s} + K_d\, s = K_p\,(1 + \frac{1}{T_i\, s} + T_d\, s)$$

in cui $T_i = K_p/K_i$ è il "tempo integrale" (o di reset) e $T_d = K_d/K_p$ è il "tempo derivativo".
I diagrammi di Bode asintotici della risposta in frequenza sono riportati in figura 5.6.
Dal momento che il controllore P.I.D. descritto in precedenza è un sistema con due zeri a parte reale negativa ed un solo polo nell'origine, esso risulta essere un sistema improprio, e pertanto non realizzabile, per la presenza del termine derivativo. In realtà, nella pratica, l'azione derivativa è ottenuta per mezzo della funzione di trasferimento:

$$R_d(s) = \frac{K_p\, T_d\, s}{1 + \frac{T_d}{N} s} = \frac{K_d\, s}{1 + \frac{K_d}{K_p\, N} s}$$

dove la costante positiva N è scelta in modo che il polo $s = -N/T_d$, aggiunto per la realizzabilità, sia all'esterno della banda di frequenze di interesse nel controllo.

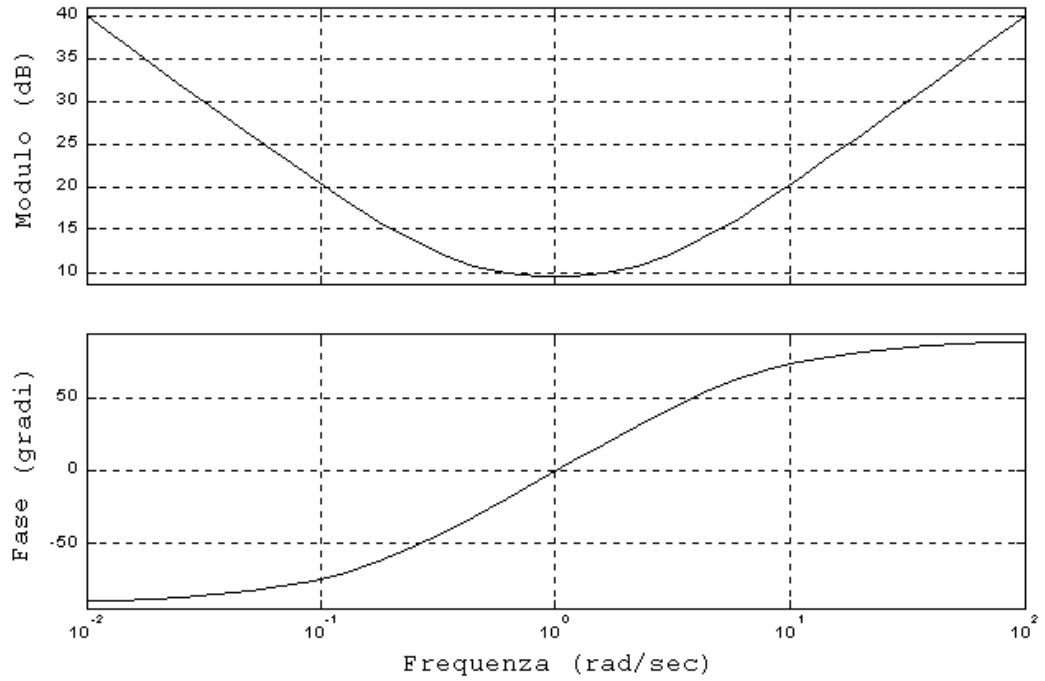

Figura 5.6: Diagrammi di Bode asintotici di un P.I.D. ideale.

5.3.2 Il fenomeno del "wind-up"

La presenza combinata dell'azione integrale e di una saturazione dovuta all'attuatore provoca un effetto di tipo non lineare che può deteriorare significativamente le prestazioni del sistema di controllo. La saturazione dell'attuatore può essere descritta dalla seguente relazione, dove u è l'ingresso ed m l'uscita:

$$m(t) = \begin{cases} -u_M & , \quad u(t) < -u_M \\ u(t) & , \quad |u(t)| \leq u_M \\ u_M & , \quad u(t) > u_M \end{cases}$$

Quando l'errore si mantiene dello stesso segno per un certo intervallo di tempo, lo stato dell'integratore cresce in modulo. Ciò avviene anche se l'effettiva variabile di ingresso m del sistema sotto controllo viene limitata al valore u_M o $-u_M$ dalla saturazione dovuta all'attuatore. Quando questo accade, se l'errore cambia segno, è necessario attendere che lo stato u dell'integratore torni ad assumere valori in modulo inferiori ad u_M prima che l'attuatore riprenda ad operare in zona lineare, cioè si abbia $m(t) = u(t)$. In altri termini si deve

attendere la scarica dell'azione integrale.
Proprio al fine di evitare tale inconveniente esistono delle tecniche di desaturazione, il cui scopo è fare in modo che la variabile di controllo effettiva lasci il valore di saturazione non appena l'errore cambia segno.
L'utilizzo di tali tecniche conduce a delle prestazioni che, in termini di sovraelongazione della risposta al gradino, sono da considerarsi perfino migliori di quelle fornite dal sistema di controllo in assenza di saturazione.

5.3.3 Taratura automatica del controllore

Quando la funzione di trasferimento del sistema sotto controllo è nota, i parametri del controllore P.I.D. possono essere tarati per mezzo di tecniche di sintesi diretta. Spesso tuttavia la determinazione di un modello a partire dalle leggi che governano i fenomeni in esame può richiedere un eccessivo impegno rispetto all'esigenza di progettare un regolatore in grado di fornire prestazioni accettabili. In questi casi è frequente l'uso di metodi automatici di taratura che consentono di pervenire direttamente alla sintesi del regolatore a partire da specifiche prove effettuate sul processo. Nella maggior parte dei casi queste prove consentono di stimare un modello matematico approssimato del sistema che sarà poi utilizzato per la sintesi del controllore.
Le prime regole di taratura furono introdotte nel 1942 da Ziegler e Nichols; tali regole si dividono in regole ad anello chiuso e regole ad anello aperto. Una volta stimato il modello matematico, è possibile, mediante l'utilizzo di opportune tabelle, determinare i parametri K_p, K_i e K_d.
Nel caso che tali metodi non forniscano una buona approssimazione del modello, è opportuno procedere mediante una sintesi per tentativi.
I valori dei parametri del controllore P.I.D. ottenuti per il controllo di livello esposto in questo capitolo sono i seguenti:

- $K_p = 50 \qquad K_i = 1 \qquad K_d = 0.5$

5.4 Modello Simulink del processo

Per poter integrare il processo nel tele-laboratorio è necessario costruire il modello Simulink che lo rappresenti; tale operazione dovrà essere compiuta dall'amministratore del tele-laboratorio, che dovrà provvedere anche all'interfacciamento con la scheda di acquisizione. Tale file dovrà essere chiamato "Source.mdl" e nel caso del controllo di livello risulterà essere come in figura 5.7. Verranno adesso descritti i vari componenti che costituiscono tale modello:

Standard Reference: questo blocco si occupa di generare il tipo di riferimento da applicare al sistema, che in questo caso può essere un gradino, una rampa od un'onda sinusoidale. Una descrizione più dettagliata di questo blocco era stata esposta nel capitolo 4.

Saturation: nel modello appaiono due blocchi di saturazione; il primo, applicato al riferimento, impedisce che questo assuma un valore al di fuori di [0,5] litri, mentre il secondo, collegato con l'uscita del controllore, si preoccupa di limitare la massima tensione applicabile alla pompa ad 8 volts.

To Workspace: questi blocchi consentono di memorizzare il corrispondente valore nel file .MAT che sarà creato al termine dell'esperimento. Non esiste un limite massimo per tali blocchi e si consiglia di inserirli in tutti quei punti che possono essere rilevanti per un'analisi a posteriori.

Out_reference, out_output, out_command: questi tre blocchi sono obbligatori e permettono di trasmettere i valori del riferimento, dell'uscita e del comando all'utente remoto.

ACT_CONTROLLER: questo blocco sarà quello preposto ad integrare il controllore definito dall'utente; al suo interno sono presenti due blocchi di ingresso ed uno di uscita, senza alcun collegamento tra di loro. Tali blocchi rappresentano il valore dell'errore, dell'uscita e del comando da applicare, che saranno utilizzati in fase di progettazione del controllore.

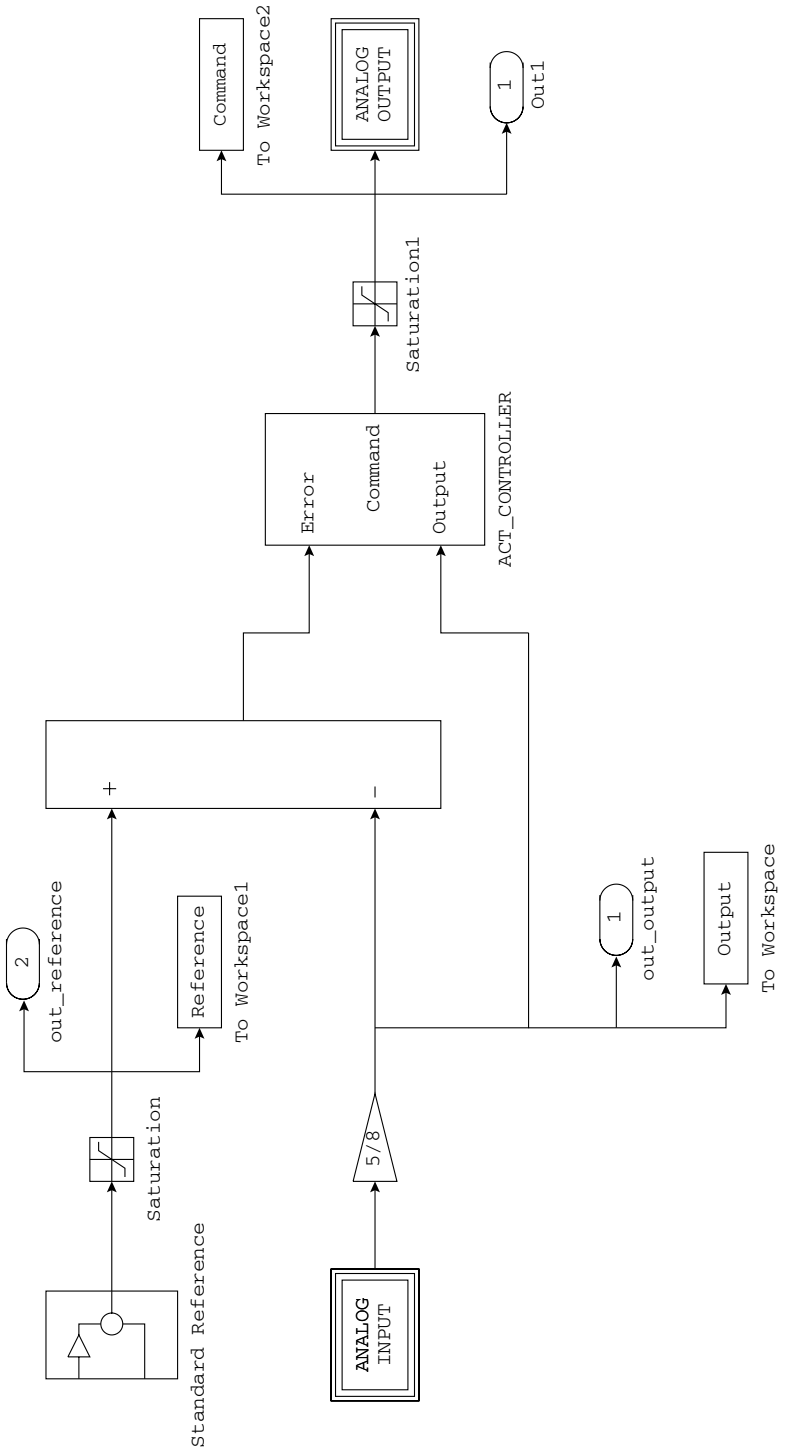

Figura 5.7: Schema Simulink del file “Source.mdl”.

Analog input, analog output: questi blocchi si preoccupano di interfacciarsi con la scheda di acquisizione dati, consentendo la lettura dell'ingresso analogico e l'impostazione di una certa tensione sull'uscita analogica. Questi componenti faranno riferimento a due S-functions predisposte a tali operazioni che saranno descritte in dettaglio nell'appendice B.

A questo punto è stato costruito il modello Simulink del processo già predisposto per essere integrato con il controllore.

5.5 Modello Simulink del controllore

Una volta progettata la legge di controllo da utilizzare, è necessario da parte dell'utente costruire un modello Simulink che la rappresenti. L'unica convenzione affinché tale modello possa essere integrato nel tele-laboratorio è quella di racchiudere il tutto in un unico sottosistema (subsystem), in cui il primo ingresso corrisponderà all'errore ed il secondo all'uscita.
Lo schema Simulink di un controllore P.I.D. progettato per il controllo di livello è rappresentato in figura 5.8, in cui il valore relativo all'uscita non viene utilizzato. La conoscenza di tale valore può comunque essere utile per altri tipi di controllo, che facciano uso, ad esempio, di una linearizzazione in retroazione, come illustrato in figura 5.9. In entrambi i casi viene sommata in uscita una tensione costante di 3.7 volts al fine di eliminare la soglia presente nella pompa.

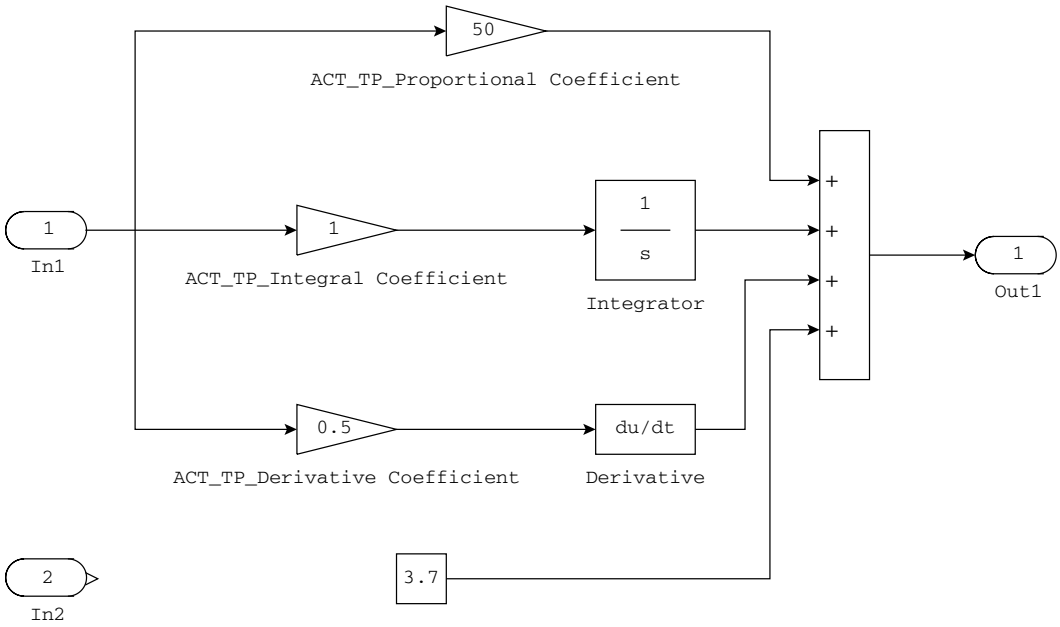

Figura 5.8: Schema Simulink di controllore P.I.D.

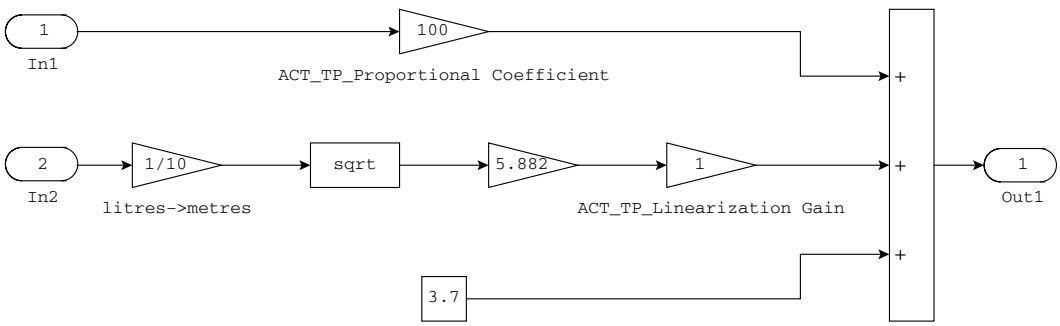

Figura 5.9: Schema Simulink di controllore con linearizzazione in retroazione.

5.6 Risultati sperimentali

Vengono di seguito riportati i grafici relativi ad alcuni esperimenti effettuati sul processo di controllo di livello.
I controllori utilizzati sono un P.I.D. (Fig. 5.8) ed un regolatore basato sulla linearizzazione in retroazione con un'azione proporzionale sull'errore (Fig. 5.9).
Gli esperimenti si riferiscono rispettivamente ad un inseguimento al gradino, alla rampa e ad un'onda sinusoidale.
Per ogni esperimento vengono riportati i grafici:

- Riferimento/Uscita: in tale grafico è possibile valutare la bontà della legge di controllo utilizzata, in quanto vengono rappresentate queste due grandezze sullo stesso diagramma.

- Comando: in questo grafico viene riportata l'uscita del controllore a cui è stata applicata una saturazione al fine di evitare rotture o malfunzionamenti di apparecchiature.

Il confronto tra i due controllori evidenzierà che quello basato sulla linearizzazione in retroazione ha delle prestazioni decisamente superiori al P.I.D., specialmente per quanto riguarda l'inseguimento al gradino. D'altronde bisogna tenere conto della maggiore facilità con cui può essere realizzato un controllore di tipo P.I.D., che molte volte può garantire un andamento del sistema controllato più che soddisfacente.

5.6.1 Controllo P.I.D.

Inseguimento di un gradino

Verranno di seguito commentati i grafici relativi all'inseguimento di un gradino di 1 litro (Fig. 5.10 e 5.11). Da tali grafici si possono valutare alcune caratteristiche del sistema controllato quali:

- Tempo di salita (10% - 90%) = 25 secondi.

- Tempo di assestamento (banda al 5% del regime) = 121 secondi.

- Sovraelongazione massima = 258 millilitri = 25.8%.

Risultano poco soddisfacenti sia il tempo di assestamento che la massima sovraelongazione, che supera addirittura il 25%. Tali mediocri prestazioni sono dovute per lo più al precedentemente descritto fenomeno del "wind-up", in quanto il controllore P.I.D. utilizzato in questo contesto non contiene blocchi di desaturazione.

Inseguimento di una rampa lineare

Viene adesso valutata la risposta relativa all'inseguimento di una rampa lineare con pendenza di 5 millilitri al secondo (Fig. 5.12 e 5.13). Tale pendenza, pur non essendo molto pronunciata, è comunque prossima al limite fisico raggiungibile dall'attuatore, per cui non avrebbe senso sceglierne una molto più ripida.
In questo caso si può notare un leggero errore di inseguimento del processo controllato, che a regime risulta comunque essere nell'ordine dei millilitri.

Inseguimento di un'onda sinusoidale

La risposta ad un'onda sinusoidale centrata in mezzo litro, con ampiezza 250 millilitri e periodo 100 secondi, può essere considerata soddisfacente dopo un certo periodo di transitorio, come è possibile notare dai grafici ad essa relativi (Fig. 5.14 e 5.15). Bisogna comunque notare che l'errore di inseguimento oscilla con punte dell'ordine dei centilitri.

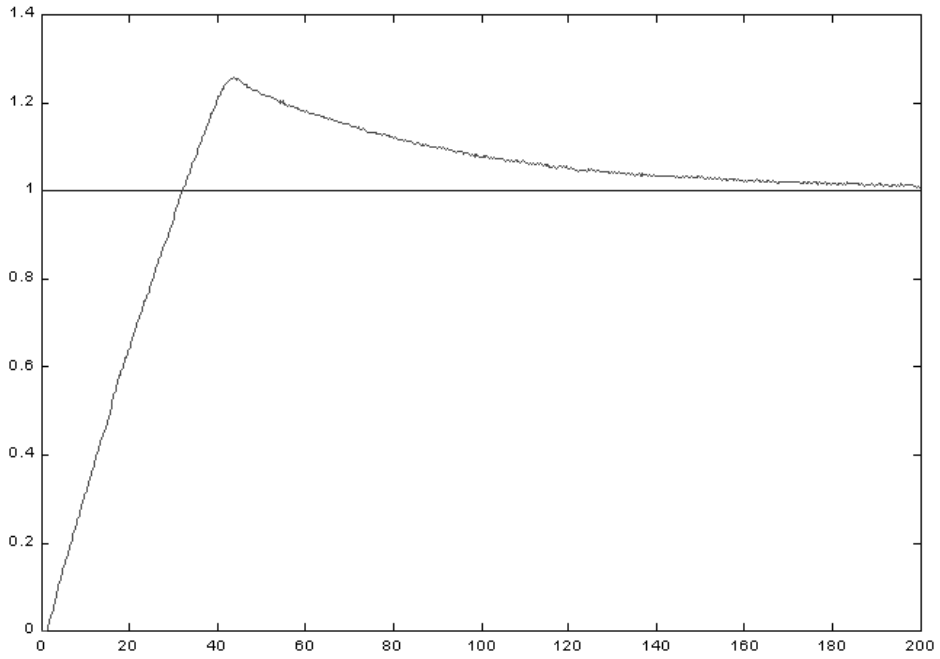

Figura 5.10: Inseguimento di un gradino (riferimento ed uscita) con controllore P.I.D.

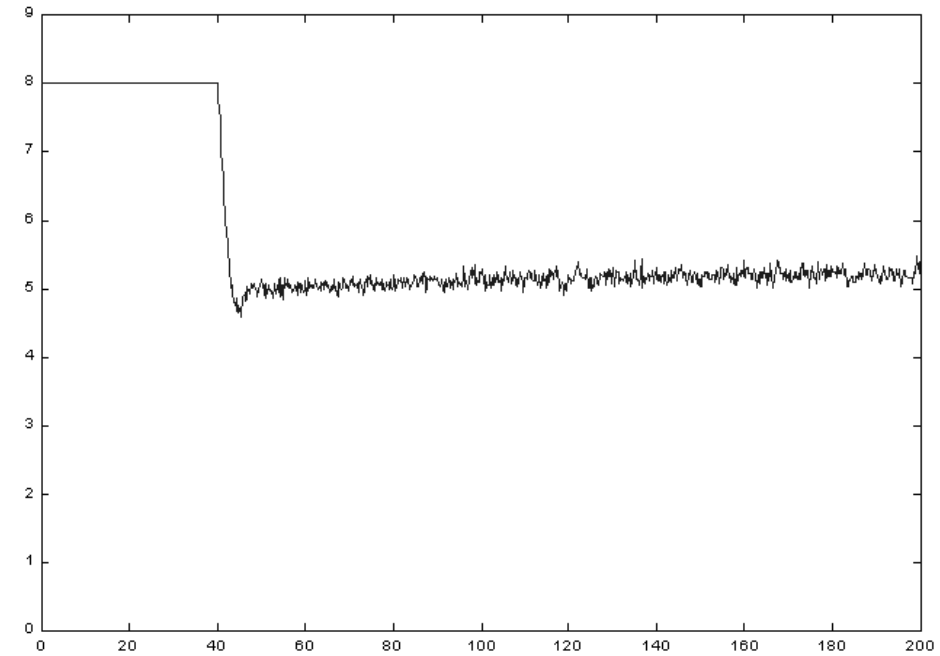

Figura 5.11: Inseguimento di un gradino (comando) con controllore P.I.D.

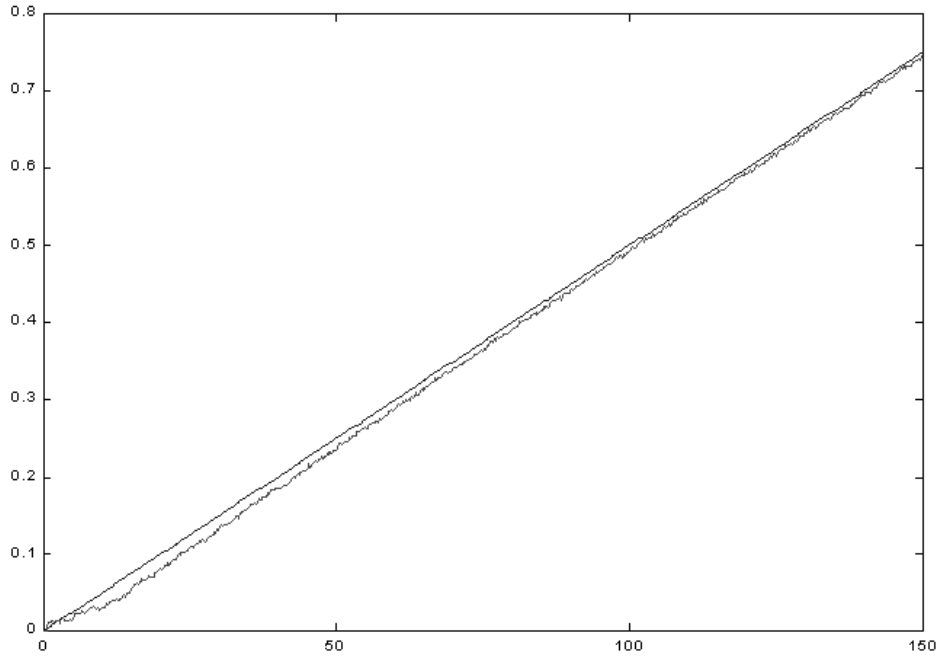

Figura 5.12: Inseguimento di una rampa lineare (riferimento ed uscita) con controllore P.I.D.

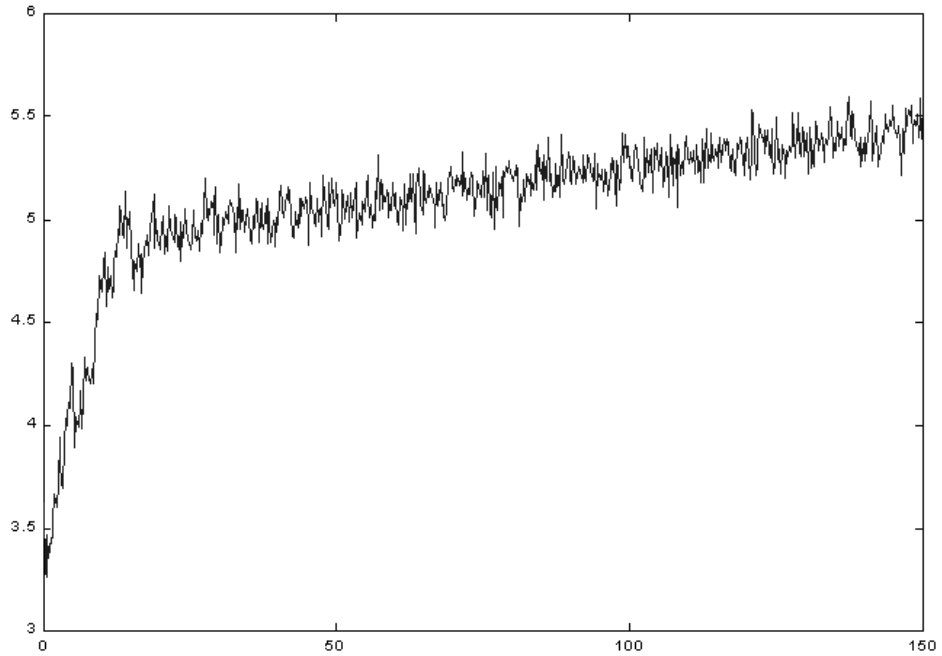

Figura 5.13: Inseguimento di una rampa lineare (comando) con controllore P.I.D.

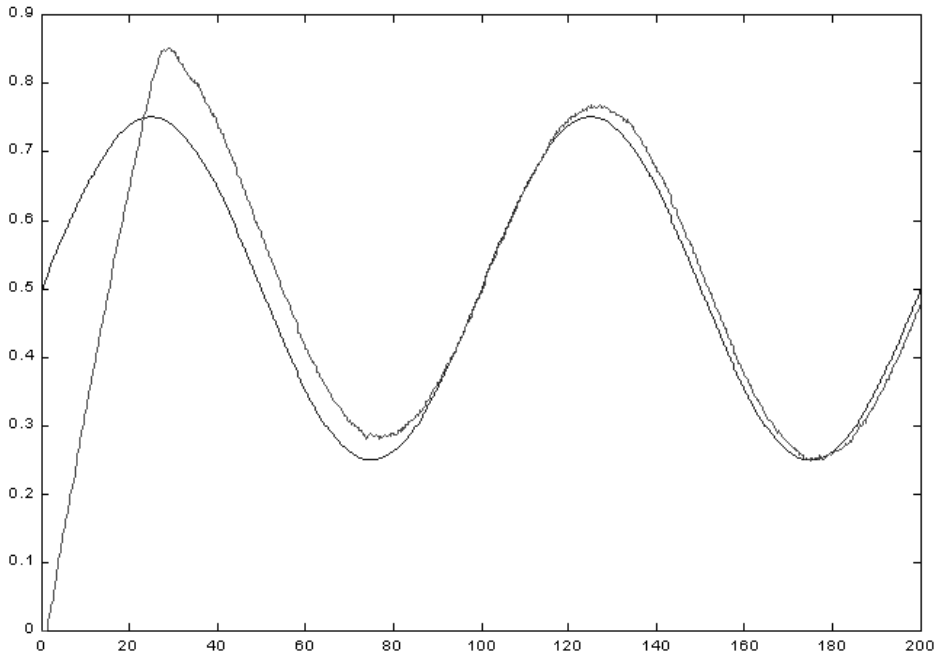

Figura 5.14: Inseguimento di una sinusoide (riferimento ed uscita) con controllore P.I.D.

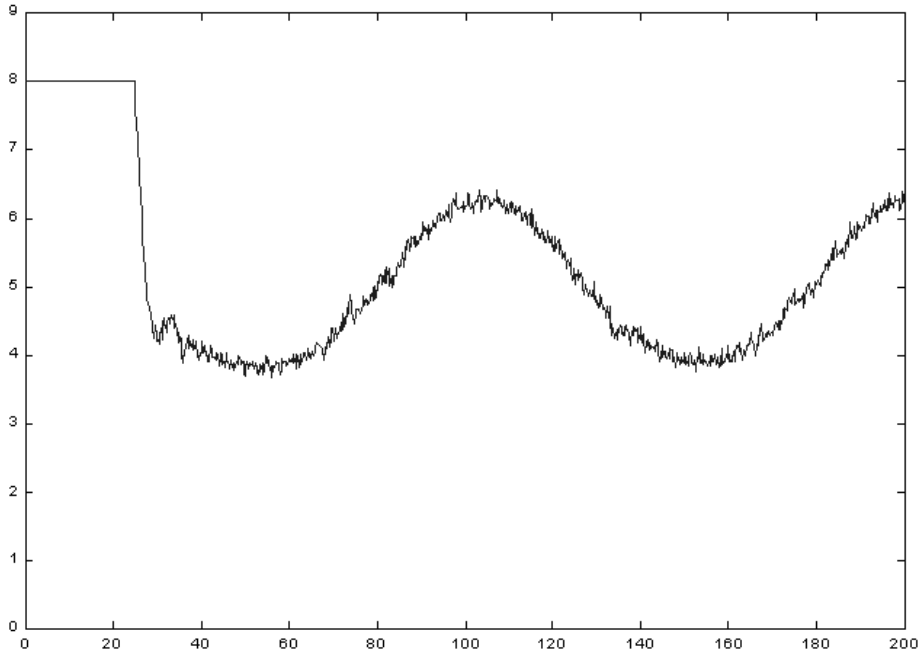

Figura 5.15: Inseguimento di una sinusoide (comando) con controllore P.I.D.

5.6.2 Controllo con linearizzazione in retroazione

Mediante tale tecnica di controllo ci proponiamo di eliminare la componente che causa la non-linearità del sistema per mezzo di un opportuno comando basato sul valore dell'uscita.
Il modello matematico del sistema, come visto in precedenza, è il seguente:

$$\dot{h} = -k\sqrt{h} + q/A_1$$

dove:

$$q = \begin{cases} 0 & \text{se } V \leq 3.7 \\ -5.032 \cdot 10^{-5} + 1.36 \cdot 10^{-5}\, V & \text{se } V > 3.7 \end{cases}$$

Applicando all'attuatore una tensione costante di 3.7 volts al fine di eliminare la soglia, otteniamo:

$$q = 1.36 \cdot 10^{-5}\, \overline{V} = p\, \overline{V}$$

con

$$\overline{V} = V - 3.7$$

e quindi:

$$\dot{h} = -k\sqrt{h} + \frac{p}{A_1}\, \overline{V}$$

Ponendo:

$$\overline{V} = k\, \frac{A_1}{p}\, \sqrt{h} + \frac{A_1}{p}\, W$$

si ottiene:

$$\dot{h} = -k\sqrt{h} + \frac{p}{A_1}\, (k\, \frac{A_1}{p}\, \sqrt{h} + \frac{A_1}{p}\, W) = W$$

A questo punto è sufficiente porre W proporzionale all'errore (tramite un coefficiente K_p) per ottenere degli ottimi risultati in termini di risposta al gradino, alla rampa ed alla sinusoide.

$$\dot{h}(t) = K_p\, e(t)$$

Inseguimento di un gradino

L'inseguimento dello stesso gradino utilizzato nel caso del controllore P.I.D. ha evidenziato le seguenti caratteristiche (Fig. 5.16 e 5.17):

- Tempo di salita (10% - 90%) = 25 secondi.
- Tempo di assestamento (banda al 5% del regime) = 31 secondi.
- Sovraelongazione massima = 18 millilitri = 1.8%.

In questo caso il sistema risulta controllato in modo ottimale, avendo una piccolissima sovraelongazione e dei tempi di salita e di assestamento minimi. Bisogna infatti ricordare che il processo in esame è particolarmente lento, vista la presenza della saturazione sull'attuatore, per cui non è possibile scendere molto al di sotto di tali tempi.

Inseguimento di una rampa e di un'onda sinusoidale

Anche per quanto riguarda l'inseguimento di una rampa (Fig. 5.18 e 5.19) e di un'onda sinusoidale (Fig. 5.20 e 5.21) i risultati ottenuti con questa tecnica di controllo risultano migliori rispetto a quelli di un controllore di tipo P.I.D., seppure in modo meno evidente rispetto al gradino.

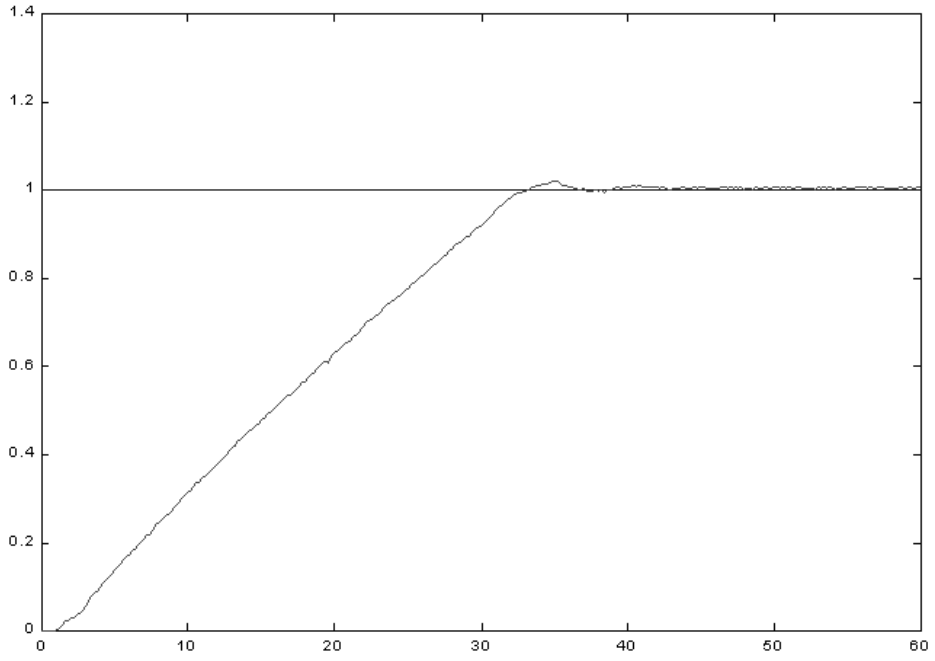

Figura 5.16: Inseguimento di un gradino (riferimento ed uscita) mediante controllore con linearizzazione in retroazione.

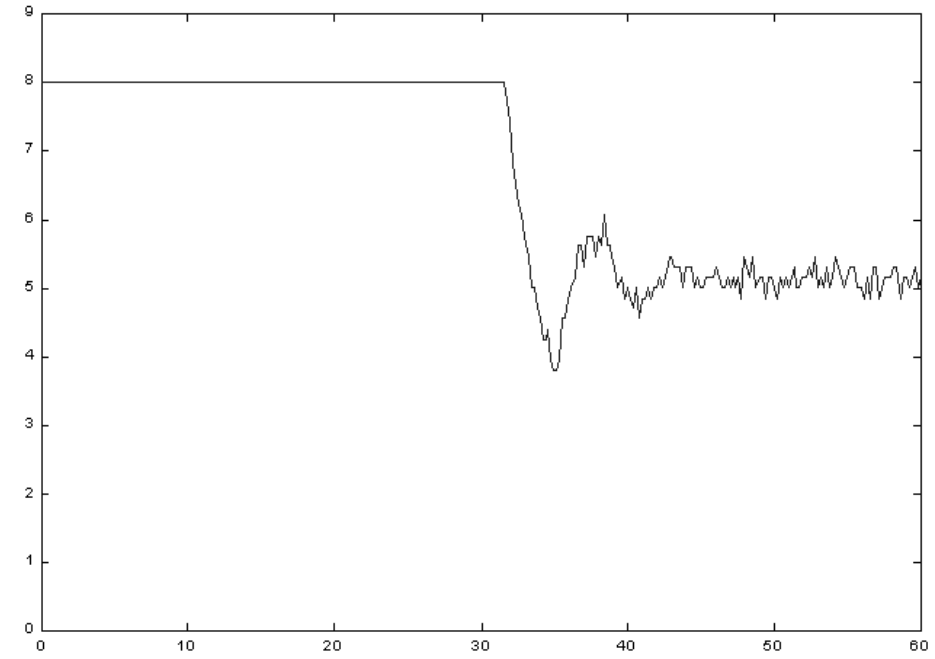

Figura 5.17: Inseguimento di un gradino (comando) mediante controllore con linearizzazione in retroazione.

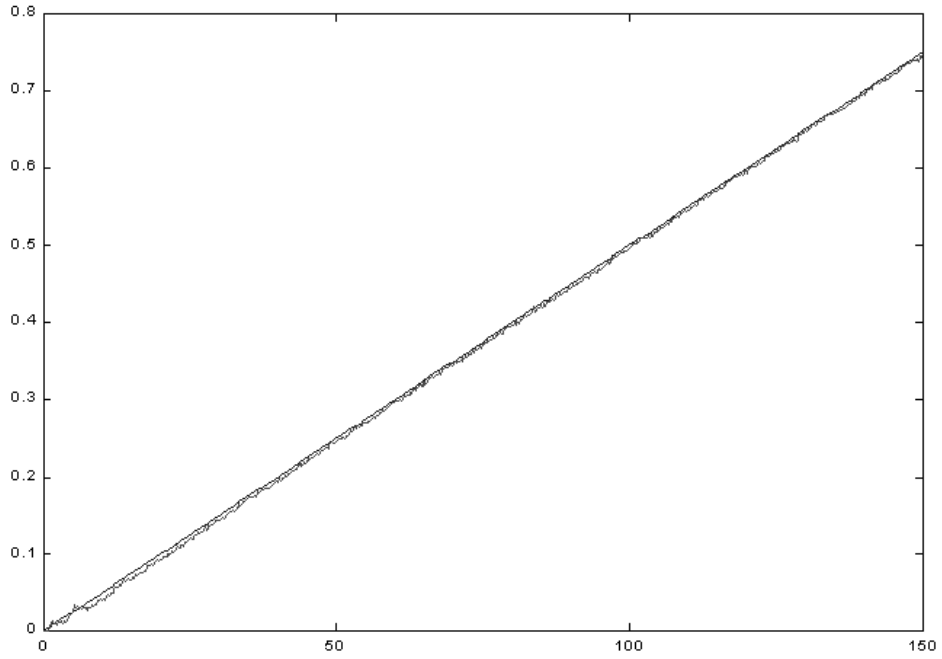

Figura 5.18: Inseguimento di una rampa lineare (riferimento ed uscita) mediante controllore con linearizzazione in retroazione.

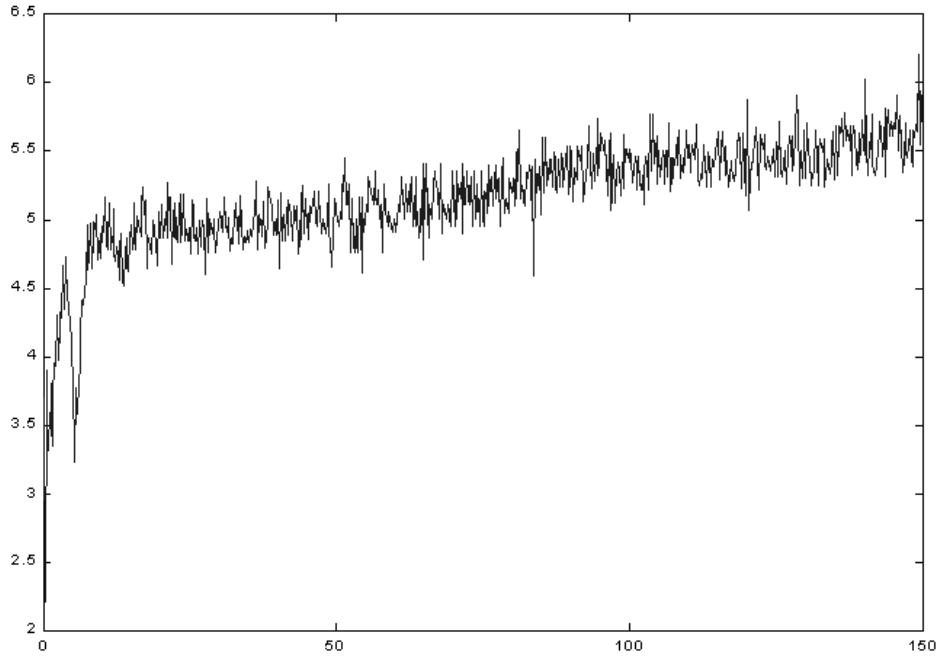

Figura 5.19: Inseguimento di una rampa lineare (comando) mediante controllore con linearizzazione in retroazione.

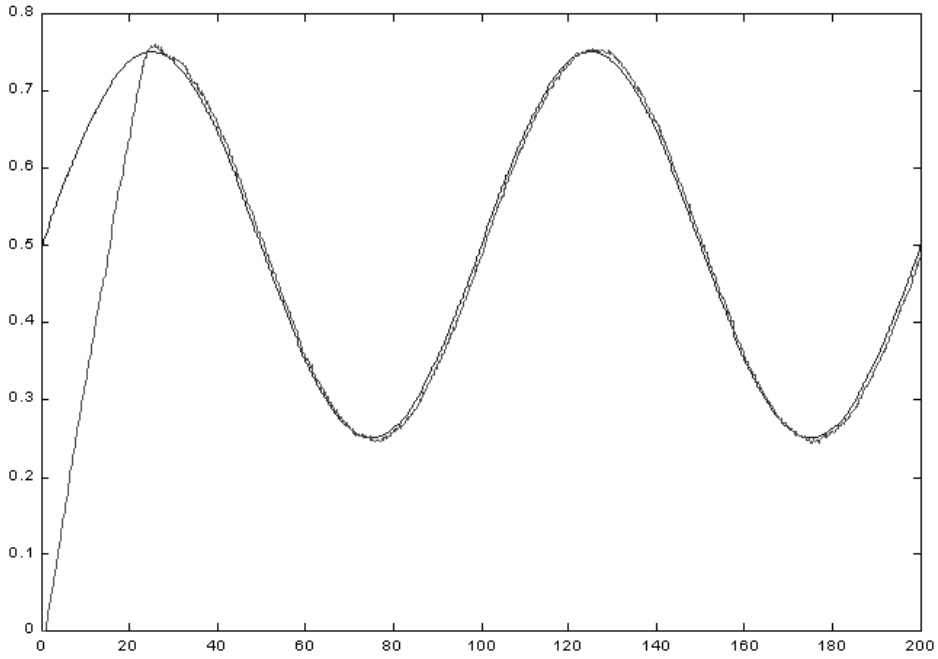

Figura 5.20: Inseguimento di una sinusoide (riferimento ed uscita) mediante controllore con linearizzazione in retroazione.

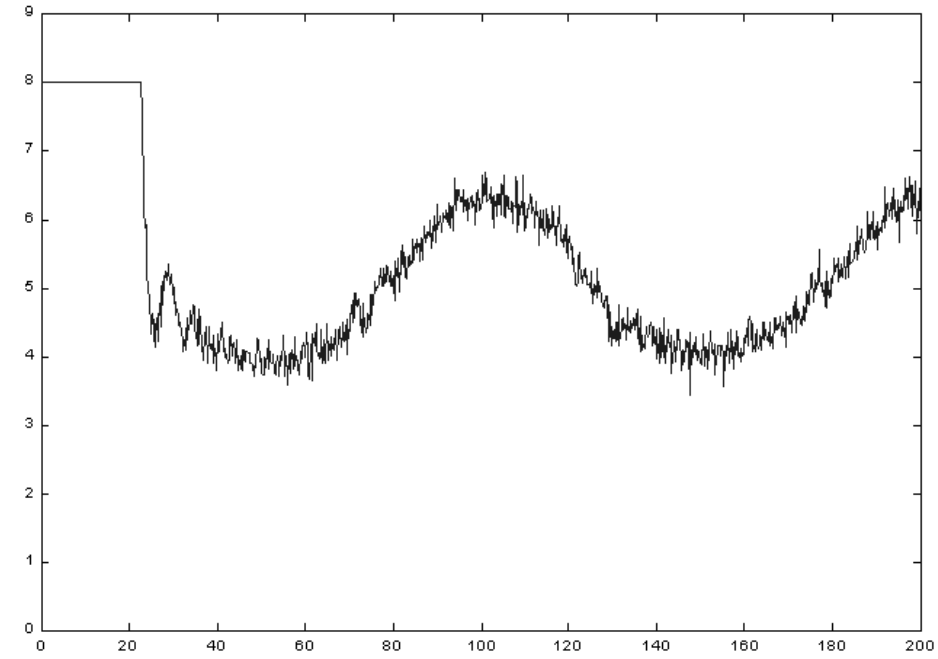

Figura 5.21: Inseguimento di una sinusoide (comando) mediante controllore con linearizzazione in retroazione.

Capitolo 6

Conclusioni e sviluppi futuri

In questo lavoro di tesi è stato analizzato e progettato un meccanismo di controllo in remoto di processi dinamici. La struttura utilizzata ha permesso di poter collegare al tele-laboratorio un numero arbitrario di processi, ognuno dei quali dovrà essere interfacciato con un calcolatore ad esso dedicato. Dal punto di vista applicativo è stato fatto uso del pacchetto Matlab, mediante il quale è possibile progettare dei controllori in modo visuale, attraverso l'interfaccia fornita da Simulink; questa scelta è stata motivata dalla presenza di vari toolbox che hanno già implementato al loro interno sofisticate leggi di controllo, consentendo quindi una rapida ed efficiente progettazione del controllore da parte dell'utente.
Dal lavoro svolto sono emerse diverse prospettive di ampliamento delle funzionalità del tele-laboratorio; le principali innovazioni che possono essere realizzate nel futuro sono elencate di seguito:

Identificazione di processi.

Il lavoro fin qui svolto si è incentrato sul controllo di un processo di cui era già noto (o direttamente ricavabile) il modello matematico; un aspetto che potrebbe essere inserito è invece quello dell'identificazione di un processo in remoto, ossia della costruzione del modello matematico a partire da una conoscenza di massima del sistema. Per poter realizzare tale operazione è necessario fornire al processo degli opportuni ingressi e di studiarne le uscite ottenute; anche in questo caso Matlab mette a

disposizione un toolbox specifico per questo tipo di operazioni. Tramite queste analisi sarà quindi possibile calcolare un modello del sistema, che potrà essere utilizzato per lo studio di una legge di controllo; al termine di queste operazioni non resta altro che spedire il controllore e valutare la sua efficienza sul processo reale.

Competizione tra studenti.

Dal punto di vista didattico potrebbe essere inserita una nuova caratteristica, che consiste in una competizione tra studenti al fine di valutare quali sono i migliori controllori da essi progettati. In questo modo verrebbe sviluppato l'interesse degli studenti stessi nei confronti della disciplina, che sarebbe inoltre appresa in maniera più completa, in quanto verrebbe adottato un vero e proprio processo reale. Il confronto tra i vari controllori analizzati dal punto di vista teorico e pratico consentirebbe inoltre una migliore comprensione di certi concetti che altrimenti potrebbero rimanere astratti.

Per poter realizzare tutto questo sarebbe necessario disporre di una nuova funzionalità, che consentisse di applicare degli ingressi prefissati in modo automatico; in tal caso, tutti i controllori verrebbero confrontati sullo stesso insieme di ingressi. A questo punto si renderebbe necessaria una procedura che calcolasse, in base alla dinamica del processo, gli indici di prestazione più significativi (tempo di salita e di assestamento, sovraelongazione massima, banda passante, ecc...), al fine di poter stilare una classifica tra i vari controllori.

Perfezionamento del feedback visivo.

Per quanto riguarda l'aspetto legato alla telecamera, è stato adottato un apposito software per videoconferenza al fine di rendere possibile la visualizzazione del processo in remoto. Tale aspetto può essere migliorato integrando la finestra preposta alla visualizzazione dell'esperimento all'interno dell'applet Java, evitando così all'utente l'onere di dover "scaricare" ed eseguire il relativo software. Oltre a questo vantaggio, tale

soluzione eviterebbe i problemi di incompatibilità presenti tra le varie piattaforme non dovendo eseguire alcun programma esterno.

Appendice A

Istruzioni di installazione

In questa appendice vengono riportate le varie operazioni che devono essere eseguite per aggiungere un nuovo processo al tele-laboratorio. Non verrà presa in considerazione la gestione della telecamera in quanto per il suo funzionamento è sufficiente l'utilizzo di un qualsiasi software per videoconferenza.

A.1 Requisiti minimi

Il calcolatore preposto al controllo del processo dovrà avere una potenza di calcolo sufficiente per poter gestire il tempo reale, e dovrà contenere le seguenti componenti hardware e software:

- Sistema operativo Windows 95/98 della Microsoft.
- Matlab versione 5.2 o successiva, con i seguenti toolbox:
  - Simulink.
  - Real-Time Workshop.
  - Control system toolbox (opzionale).
  - Robust control toolbox (opzionale).
  - Fuzzy logic toolbox (opzionale).
- Compilatore Watcom C/C++ versione 11.0.

- Programma di gestione di un server web.
- Scheda di acquisizione dati A/D e D/A.
- Librerie in linguaggio C relative alla scheda di acquisizione.
- Modem o connessione diretta alla rete Internet.

Alcuni toolbox di Matlab sono stati indicati come opzionali, in quanto la loro presenza non influisce sul funzionamento generale del tele-laboratorio; bisogna però ricordare che una loro eventuale assenza non consentirà di progettare controllori contenenti i blocchi inclusi in tali toolbox, per cui la loro installazione viene vivamente consigliata.
Per quanto riguarda le directory in cui saranno installati i vari programmi, viene lasciata piena libertà di scelta. Per semplificare la trattazione supporremo comunque che le directory utilizzate siano le seguenti:

- C:\MATLAB per quanto riguarda il pacchetto Matlab.
- C:\WATCOM relativamente al compilatore C.
- C:\ACQ\INCLUDE e C:\ACQ\LIB per quanto concerne le librerie della scheda di acquisizione.

A.2 Installazione del software

Per installare il software specifico del tele-laboratorio è necessario scompattare il file "Act_process.zip" in una qualsiasi directory dell'hard disk, che supponiamo essere C:\ACT_PROCESS.
L'albero delle directory risultante sarà quello evidenziato in figura A.1. Nella directory ACT_PROCESS sarà presente il file Interface.exe, che, nel caso si desideri avviare il gestore del tele-laboratorio all'accensione del calcolatore, dovrà essere inserito tra i programmi in "esecuzione automatica" di Windows; lo stesso dicasi per il software relativo al server web, qualunque esso sia. È inoltre consigliabile la rimozione di tutti gli altri programmi dalla cartella "esecuzione automatica" di Windows, in quanto potrebbero ridurre le prestazioni

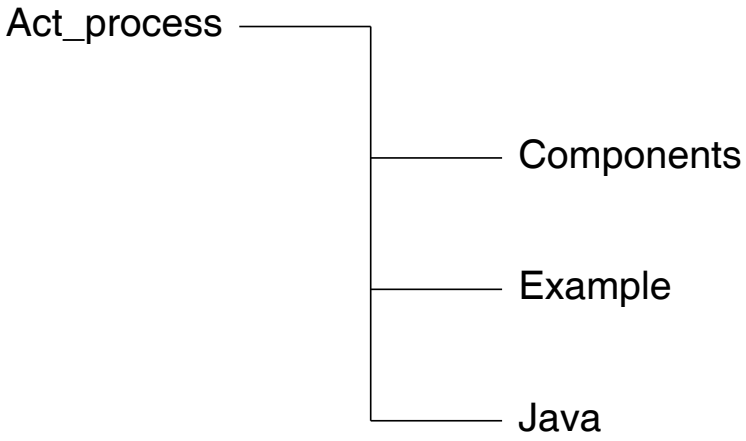

Figura A.1: Albero delle directory del Tele-Laboratorio.

della gestione in tempo reale.
Oltre ad Interface.exe la directory ACT_PROCESS conterrà tutti i files necessari al controllo del processo, nonché i files .OBJ che verranno creati in fase di compilazione.
All'interno della directory COMPONENTS risiederanno quei files che dovranno essere copiati in opportune directories come descritto in seguito.
Nella directory EXAMPLE saranno presenti alcuni files che potranno essere utilizzati come guida per la costruzione dei modelli Simulink e di altri files necessari per il corretto funzionamento del tele-laboratorio.
La directory JAVA conterrà infine le due applets Telelab1 e Telelab2 con incluse le altri classi indispensabili.
Al fine di rendere operativo il tele-laboratorio dovranno essere eseguite le seguenti procedure:

- Copiare il file Grt.tlc nella directory \MATLAB\RTW\C\GRT.
- Copiare il file Pt_Print.c in \MATLAB\RTW\C\SRC.
- Modificare il file Grt_watc.tmf togliendo i commenti nelle righe evidenziate da '*'; tali linee indicano i percorsi dove andare a recuperare le librerie relative alla scheda di acquisizione. Nel caso venisse usata la scheda Ni-

daq PC-1200, tali righe andrebbero modificate nel seguente modo:

USER_INCLUDES=-Ic:\acq\include

USER_LIBS=-Ic:\acq\lib\Nidaq.lib

Una volta effettuate tali modifiche copiare il file nella directory:
\MATLAB\RTW\C\GRT

- Impostare la directory C:\WATCOM come quella predefinita per la compilazione sotto Matlab; per fare ciò eseguire in ambiente Matlab il comando:

  mex -setup

  e seguire le istruzioni.

- Scrivere e compilare i blocchi Simulink preposti all'interfacciamento con la scheda di acquisizione (vedi appendice B), e copiare i files .C e .DLL ad essi relativi nella directory \ACT_PROCESS.

- Scrivere il file Source.mdl e copiarlo nella directory \ACT_PROCESS. Un esempio di tale file è presente nella sottodirectory \EXAMPLE.

- Modificare il file Interface.prf, contenente i percorsi e le preferenze utilizzate, quali ad esempio la durata dei time-outs.

- Impostare come pagina predefinita del server web:

  C:\ACT_PROCESS\JAVA\Telelab1.htm

- Per impostare dei controllori come predefiniti è necessario modificare il file Process.dat inserendo il titolo del processo ed i nomi dei vari controllori predefiniti (con le relative directory). Anche in questo caso è possibile seguire la traccia presente nella directory degli esempi.
  Dovranno poi essere copiati i files Process.exe ottenuti nelle directories relative ai controllori realizzati.

Appendice B

Interfaccia con la scheda di acquisizione

Affinché un processo possa essere integrato in un modello Simulink è necessario costruire dei blocchi che avranno lo scopo di interfacciarsi con la scheda di acquisizione dati, al fine di poter ricevere i segnali dai trasduttori e di poter inviare il comando agli attuatori. Per fare questo, Simulink mette a disposizione delle speciali funzioni chiamate S-functions (System functions) [4], che permettono di aggiungere i propri algoritmi ad un modello Simulink.
Tali S-functions dovranno essere scritte in linguaggio C e compilate attraverso un opportuno comando di Matlab.

B.1 Descrizione delle operazioni necessarie

Per costruire un blocco preposto all'interfacciamento con il processo è necessario compiere le operazioni sottoelencate:

- Scrivere una S-function, secondo le convenzioni che verranno descritte in seguito; ipotizziamo che tale funzione abbia lo scopo di acquisire il valore presente sull'ingresso analogico di una scheda di acquisizione, che supporremo essere del tipo Nidaq PC-1200 della National Instruments, in modo da poter fornire un esempio concreto per la realizzazione di tale blocco. Tale file sarà chiamato get_input.c.

- Compilare il file precedentemente creato eseguendo il seguente comando dal workspace di Matlab:
  mex get_input.c -I\acq\include \acq\lib\nidex.lib \acq\lib\nidaq.lib
- Inserire nel modello Simulink un blocco del tipo "S-function", appartenente alla libreria "Nonlinear". Inserire nel campo relativo al nome il file in questione, ossia get_input.
- (opzionale) Mascherare il blocco precedentemente ottenuto al fine di facilitarne l'utilizzo.

A questo punto il blocco ottenuto può essere utilizzato ogni qual volta si renda necessario acquisire il valore dell'ingresso analogico della scheda di acquisizione.

B.2 Implementazione del blocco "get_input"

Verrà di seguito esposto l'algoritmo utilizzato per ottenere il file get_input.c, che dovrà realizzare l'interfacciamento con un ingresso analogico della scheda di acquisizione. Tale file potrà essere facilmente adattato nel caso lo si voglia utilizzare per leggere un contatore o un ingresso digitale.

1. *#define S_FUNCTION_LEVEL 2*
2. *#define S_FUNCTION_NAME get_input*
3. *#include simstruc.h*
4. *#include Nidaqex.h*
5. *#define PARAM1(S) ssGetSFcnParam(S,0)*
6. *#define PARAM2(S) ssGetSFcnParam(S,1)*
7. *#if defined(MATLAB_MEX_FILE)*
   - 7.1. *#define MDL_CHECK_PARAMETERS*
   - 7.2. *static void mdlCheckParameters(SimStruct ∗S)*
     {
     - 7.2.1. *if (mxGetNumberOfElements(PARAM1(S)) != 1)* {
       - 7.2.1.1. *ssSetErrorStatus(S,Il primo parametro deve essere*

uno scalare! (N° device));

7.2.1.2. return;}

7.2.2. else if (mxGetPr(PARAM1(S))[0] <= 0) {

7.2.2.1. ssSetErrorStatus(S, Il primo parametro deve essere positivo! (N° device));

7.2.2.2. return;}

7.2.3. if (mxGetNumberOfElements(PARAM2(S)) != 1) {

7.2.3.1. ssSetErrorStatus(S,Il secondo parametro deve essere uno scalare! (N° analog input));

7.2.3.2. return;}

7.2.4. else if ((mxGetPr(PARAM2(S))[0] < 0) || (mxGetPr(PARAM2(S))[0] > 7)){

7.2.4.1. ssSetErrorStatus(S, Il secondo parametro deve essere compreso in [0,7] (N° analog input));

7.2.4.2. return;}

}

8. #endif

*9. static void mdlInitializeSizes(SimStruct *S)*

{

9.1. ssSetNumSFcnParams(S,2);

9.2. #if defined(MATLAB_MEX_FILE)

9.2.1. if (ssGetNumSFcnParams(S) == ssGetSFcnParamsCount(S)) {

9.2.1.1. mdlCheckParameters(S);

9.2.1.2. if (ssGetErrorStatus(S) != NULL) return; }

9.2.2. else return;

9.3. #endif

9.4. ssSetNumContStates(S,0);

9.5. ssSetNumDiscStates(S,0);

9.6. if (!ssSetNumInputPorts(S,0)) return;

9.7. if (!ssSetNumOutputPorts(S,1)) return;

9.8. ssSetOutputPortWidth(S,0,1);

9.9. ssSetNumSampleTimes(S,1);

9.10. ssSetNumRWork(S,0);

9.11. ssSetNumIWork(S,0);

9.12. ssSetNumPWork(S,0);

9.13. ssSetNumModes(S,0);

9.14. ssSetNumNonsampledZCs(S,0);

9.15. ssSetOptions(S,0);

}

10. static void mdlInitializeSampleTimes(SimStruct ∗S)

{

10.1. ssSetSampleTime(S,0,INHERITED_SAMPLE_TIME);

10.2. ssSetOffsetTime(S,0,0.0);

}

11. #define MDL_START

12. #if defined(MDL_START)

12.1. static void mdlStart(SimStruct ∗S) {}

13. #endif

14. static void mdlOutputs(SimStruct ∗S,int_T tid)

{

14.1. i16 iDevice = mxGetPr(PARAM1(S))[0];

14.2. i16 iChan = mxGetPr(PARAM2(S))[0];

14.3. i16 iGain = 1;

14.4. f64 Value;

14.5. real_T ∗Value = ssGetOutputPortRealSignal(S,0);

14.6. AI_VRead(iDevice,iChan,iGain,&Value);

14.7. ∗Value = ulCount;

}

15. static void mdlUpdate(SimStruct ∗S,int_T tid) {}

16. static void mdlDerivatives(SimStruct ∗S) {}

17. static void mdlTerminate(SimStruct ∗S) {}

18. #ifdef MATLAB_MEX_FILE

18.1. #include simulink.c

19. #else

19.1. #include cg_sfun.h
20. #endif

Verrà ora commentato il precedente algoritmo, soffermandosi sulle linee di programma che dovranno essere modificate per ottenere un'interfaccia con un ingresso digitale od un contatore.

1 - 3 Dichiarazioni obbligatorie.

4 Questa linea dovrà essere modificata nel caso non si usasse una scheda di acquisizione della National Instruments.

5 - 6 Definizione di due parametri della funzione. In questo caso i parametri riguarderanno il numero del dispositivo (primo parametro) ed il numero dell'ingresso analogico (secondo parametro).

7 - 8 Controllo della correttezza dei parametri. In questo esempio viene richiesto che il primo parametro sia positivo, mentre il secondo dovrà essere compreso tra 0 e 7.

9 Vengono qui definite le caratteristiche del blocco, quali il numero di ingressi, di uscite, di stati e di parametri. Nella riga 9.1 viene indicato il numero dei parametri richiesti, mentre nelle linee 9.6 e 9.7 viene fissato il numero di ingressi ed uscite del blocco. Le altre righe non necessitano generalmente di essere modificate.

10 Definizione dei tempi di campionamento del modello che non necessitano di essere variati.

11 - 13 Possono essere inserite in questa sede le istruzioni di inizializzazione, se presenti.

14 La funzione "mdlOutputs" è il vero e proprio corpo del programma in cui dovranno essere inserite le righe necessarie a realizzare l'interfacciamento con la scheda di acquisizione. Come prima cosa vengono assegnati i due parametri definiti in precedenza a due variabili di un opportuno tipo.

Nella linea 14.5 viene assegnata l'uscita del blocco alla variabile "Value", per cui il valore di tale variabile sarà quello prelevabile dal modello Simulink.

La riga 14.6 è specifica del modello della scheda di acquisizione adottata; in questo caso la funzione "AI_VRead" necessita di quattro parametri e restituisce il valore dell'ingresso analogico selezionato. Per poter effettuare letture di ingressi digitali o contatori sarà quindi sufficiente modificare questa linea, facendo però attenzione che i tipi di variabile utilizzati siano compatibili con la funzione adottata.

15 - 16 Definizioni obbligatorie.

17 Possono essere inserite qui le istruzioni che dovranno essere eseguite al termine della simulazione.

18 - 20 Definizioni obbligatorie.

B.3 Implementazione del blocco "set_output"

Viene adesso esposto l'algoritmo utilizzato per ottenere il file set_output.c, il cui scopo sarà quello di interfacciarsi con un'uscita analogica della scheda di acquisizione. Anche in questo caso, l'algoritmo potrà essere facilmente modificato al fine di adattarlo ad una uscita digitale.

1. *#define S_FUNCTION_LEVEL 2*
2. *#define S_FUNCTION_NAME set_output*
3. *#include simstruc.h*
4. *#include Nidaqex.h*
5. *#define PARAM1(S) ssGetSFcnParam(S,0)*
6. *#define PARAM2(S) ssGetSFcnParam(S,1)*
7. *#if defined(MATLAB_MEX_FILE)*
   - 7.1. *#define MDL_CHECK_PARAMETERS*
   - 7.2. *static void mdlCheckParameters(SimStruct ∗S)*

```
{
7.2.1. if (mxGetNumberOfElements(PARAM1(S)) != 1) {
   7.2.1.1. ssSetErrorStatus(S,Il primo parametro deve essere
            uno scalare! (N° device));
   7.2.1.2. return;}
7.2.2. else if (mxGetPr(PARAM1(S))[0] <= 0) {
   7.2.2.1. ssSetErrorStatus(S, Il primo parametro deve essere
            positivo! (N° device));
   7.2.2.2. return;}
7.2.3. if (mxGetNumberOfElements(PARAM2(S)) != 1) {
   7.2.3.1. ssSetErrorStatus(S,Il secondo parametro deve essere
            uno scalare! (N° analog output));
   7.2.3.2. return;}
7.2.4. else if ((mxGetPr(PARAM2(S))[0] < 0) ||
      (mxGetPr(PARAM2(S))[0] > 7)){
   7.2.4.1. ssSetErrorStatus(S, Il secondo parametro deve essere
            compreso in [0,7] (N° analog output));
   7.2.4.2. return;}
}
8. #endif
9. static void mdlInitializeSizes(SimStruct *S)
{
9.1. ssSetNumSFcnParams(S,2);
9.2. #if defined(MATLAB_MEX_FILE)
   9.2.1. if (ssGetNumSFcnParams(S) == ssGetSFcnParamsCount(S)) {
      9.2.1.1. mdlCheckParameters(S);
      9.2.1.2. if (ssGetErrorStatus(S) != NULL) return; }
   9.2.2. else return;
9.3. #endif
9.4. ssSetNumContStates(S,0);
9.5. ssSetNumDiscStates(S,0);
9.6. if (!ssSetNumInputPorts(S,1)) return;
```

```
9.7. ssSetInputPortWidth(S,0,1);
9.8. ssSetInputPortDirectFeedThrough(S,0,1);
9.9. if (!ssSetNumOutputPorts(S,0)) return;
9.10. ssSetNumSampleTimes(S,1);
9.11. ssSetNumRWork(S,0);
9.12. ssSetNumIWork(S,0);
9.13. ssSetNumPWork(S,0);
9.14. ssSetNumModes(S,0);
9.15. ssSetNumNonsampledZCs(S,0);
9.16. ssSetOptions(S,0);
}
10. static void mdlInitializeSampleTimes(SimStruct *S)
{
10.1. ssSetSampleTime(S,0,INHERITED_SAMPLE_TIME);
10.2. ssSetOffsetTime(S,0,0.0);
}
11. #define MDL_START
12. #if defined(MDL_START)
12.1. static void mdlStart(SimStruct *S) {}
13. #endif
14. static void mdlOutputs(SimStruct *S,int_T tid)
{
14.1. i16 iDevice = mxGetPr(PARAM1(S))[0];
14.2. i16 iChan = mxGetPr(PARAM2(S))[0];
14.3. f64 dVoltage;
14.4. InputRealPtrsType uPtrs = ssGetInputPortRealSignalPtrs(S,0);
14.5. dVoltage = *uPtrs[0];
14.6. AO_VWrite(iDevice, iChan, dVoltage);
}
15. static void mdlUpdate(SimStruct *S,int_T tid) {}
16. static void mdlDerivatives(SimStruct *S) {}
17. static void mdlTerminate(SimStruct *S)    {
```

17.1. int_T iDevice = mxGetPr(PARAM1(S))[0];

17.2. int_T iChan = mxGetPr(PARAM2(S))[0];

17.3. AO_VWrite(iDevice, iChan, 0.0);

}

18. #ifdef MATLAB_MEX_FILE

18.1. #include simulink.c

19. #else

19.1. #include cg_sfun.h

20. #endif

Vista la similitudine tra questo algoritmo ed il precedente, verranno di seguito descritte soltanto le principali differenze tra i due, non soffermandosi sugli aspetti già trattati.

14 Nella riga 14.4 viene assegnata ad una variabile il valore in ingresso al blocco Simulink; tale valore viene poi utilizzato nella funzione "AO_VWrite", specifica della scheda di acquisizione utilizzata per questo esempio.

17 Per motivi di sicurezza al termine della simulazione viene posto a 0 il valore dell'uscita analogica.

Bibliografia

[1] Paolo Bolzern, Riccardo Scattolini, and Nicola Schiavoni. *Fondamenti di controlli automatici*. McGraw-Hill, 1998.

[2] Rex E. Gantenbein, Thomas L. James, John R. Cowles, and William Paloski. Telelab: A virtual laboratory for acquisition and distribution of scientific data on the internet. Technical report, Department of Computer Scienze - University of Wyoming.

[3] The MathWorks Inc. *Real-Time Workshop User's Guide*. 1997.

[4] The MathWorks Inc. *Using Simulink*. 1998.

[5] Giovanni Marro. *Controlli Automatici*. Zanichelli, 1992.

[6] Patrick Naughton and Herbert Schildt. *Java - La guida completa*. McGraw Hill, 1997.

[7] ElettronicaVeneta & INEL S.p.A. *Descrizione dei moduli "G30A" e "G30B"*.

[8] W. Richard Stevens. *TCP/IP Illustrated*, volume 1. Addison-Wesley, 1994.