Prova Curricolare

Federico Rosi

Luglio 2024

1 Automazione E Robotica con Laboratorio Progetto: Robot Antropomorfo

1.1 Introduzione

Il progetto del Robot Antropomorfo consiste nella modellazione di un robot in 3D, controllo dello stesso tramite cinematica inversa e possibilita' di ruotare l'organo terminale del robot attorno ad un punto desiderato nello spazio. Questo problema è di grande interesse nel mondo della robotica, poiché è alla base dello sviluppo di braccia robotiche impiegate ad esempio nelle catene di montaggio.

1.2 Descrizione Progetto

Lo scopo di questo progetto è stato quello di studiare ed implentare nell'ambiente di sviluppo "Processing", la cinematica inversa di un robot antropomorfo. Dopo aver realizzato in 3D il robot, che consiste di un braccio robotico con una pinza terminale, abbiamo definito le funzioni per la risoluzione della cinematica inversa del robot stesso. La risoluzione del problema della cinematica inversa ci permette di calcolare il moto che il robot deve seguire per raggiungere una posizione desiderata. Un primo problema affrontato è stato studiare come funzionassero le coordinate di Processing, per poi trovare un cambio di coordinate adatto. Infatti il sistema di Processing risulta ruotato; l'asse Y è diretta verso il basso, l'asse Z è uscente, mentre l'asse X è diretta verso destra.

Per conseguire l'orientamento desiderato della pinza, abbiamo individuato l'asse z_6 della pinza mediante gli angoli di azimuth α e di elevazione β rispetto al sistema di base, (x_0, y_0, z_0) e abbiamo definito x_6 e y_6 facendo seguire una rotazione di un angolo θ intorno all'asse z_6. Risolta poi la cinematica inversa, abbiamo ottenuto gli angoli di rotazione dei vari giunti del braccio robotico, che sono stati utilizzati per animare il modello 3D attraverso la funzione `rotate()`. Successivamente, attraverso la rappresentazione di Denavit-Hartenberg, ovvero un procedimento sistematico per associare un sistema di riferimento cartesiano ortonormale a ciascun link di un robot, abbiamo trovato il vettore delle coordinate nello spazio dei giunti, che ci ha permesso infine di ottenere la posizione e l'orientamento dell'organo terminale del manipolatore rispetto a una terna di riferimento fissa; ovvero la terna centrata nella base del robot.

Questo progetto mi ha permesso di comprendere meglio tutta la teoria studiata nel corso, in particolare i cambi di coordinate, la cinematica inversa e la procedura di Denavit-Hartenberg per l'assegnazione dei sistemi di riferimento.

2 Automazione E Robotica con Laboratorio Progetto: Uniciclo

2.1 Introduzione

Il progetto del robot uniciclo consiste nel creare una legge oraria per un robot uniciclo, basata sulla stima fornita da un filtro di Kalman esteso. Il filtro di Kalman esteso, permette di realizzare una stima piuttosto accurata delle variabili di stato di un sistema dinamico, nonostante la presenza di rumore in ingresso, o disturbi sulle misure. Inoltre sarà necessario creare dei punti di riferimento o landmark, in relazione ai quali l'uniclo calcola la sua posizione nello spazio. Un altro problema deriva dalla potenziale presenza di vincoli non olonomi, ovvero vincoli che limitano l'insieme delle direzioni accetabili per le velocità, ma non per quanto riguarda le posizioni ammisibili.

2.2 Discussione progetto

Per implementare il progetto, abbiamo modificato un esercizio studiato a lezione. Una delle principali difficoltà è stata la modifica delle informazioni utilizzate per implementare la stima definita dal filtro di Kalman esteso nel passo di correzione. Infatti invece di stimare la posizione del robot in funzione della distanza da un landmark, abbiamo utilizzato l'angolo di bearing, che rappresenta l'angolo tra la congiunte landamrk-robot e la direzione del robot stesso.

Un altro punto del progetto era implementare un campo visivo per l'uniciclo, e la possibilità di creare, accendere o spegnere svariati landmark. In particolare abbiamo rappresentato il campo visivo come un settore circolare di cui è possibile variare raggio e angolo. Inoltre abbiamo creato delle variabili booleane per tenere traccia di quali landmark fossero spenti o accesi, e abbiamo misurato come all'aumentare di quest'ultimi, la stima migliorasse notevolmente.

Questo progetto mi ha permesso di toccare con mano come funzionano realmente i sistemi dinamici, e gli algoritmi di stima. La parte più interessante del progetto a parer mio, è stata analizzare la stima al variare dei landmark, e vedere effettivamente come con minori informazioni l'uniciclo adottatava traiettori errate.

3 Automazione E Robotica con Laboratorio Progetto: PLC Siemens

3.1 Introduzione

I PLC Siemens sono apparati elettronici programmabili utilizzati per controllare e automatizzare processi industriali. Sono progettati per gestire una vasta gamma di applicazioni, dalla produzione all'automazione delle macchine, fino agli impianti di produzione industriali più complessi. I PLC Siemens vengono programmati attraverso un linguaggio chiamato STEP 7, che è ampiamente utilizzato nell'automazione industriale. Lo scopo di questo progetto è simulare un processo industriale costituito da una macchina che lavora su tre tipi di parti. I pezzi attendono di essere lavorati in tre buffer distinti, ognuno con una capacità massima limitata.

3.2 Discussione progetto

Come primo punto, abbiamo soddisfatto le specifiche grafiche richieste, che prevedevano di variare il colore della macchina in base al suo stato. Abbiamo quindi impostato il colore della macchina per

corrispondere al colore del buffer su cui sta lavorando. Inoltre, abbiamo stabilito che la macchina si colorasse di nero quando inattiva e di verde durante il setup. Il tempo di setup è molto importante perché permette alla stessa macchina di lavorare su tipi diversi di parti, in particolare nel mondo reale, cambiando ad esempio organo terminale, posizione o strumento di assemblaggio. Successivamente, abbiamo modificato il pannello HMI, un pannello grafico che consente di visualizzare in tempo reale l'evoluzione dinamica del contenuto dei tre buffer. Abbiamo inserito tre interruttori associati a ciascuno dei tre buffer, che permettono di disattivare gli arrivi, ovvero smettere di rifornire il buffer. Abbiamo anche inserito dei campi di input/output per leggere e modificare i valori dei buffer. Il principale obiettivo del progetto è stata modificare il processo di lavorazione dei buffer, in modo che non fosse più effettuato manualmente, ma seguendo una politica chiamata CLB (Clear Largest Buffer First). Questa politica svuota i buffer scegliendo come successivo buffer da lavorare quello con il maggior numero di pezzi. Mentre i pezzi di tipo `3` sono 1:1 con il prodotto finale, i pezzi di tipo 1 e 2 vengono assemblati in rapporto 1:2 a formare una parte di tipo `1-2`

Per risolvere questo punto, abbiamo prima definito un diagramma funzionale sequenziale chiamato "SFC" (Sequential Function Chart), ovvero uno strumento grafico particolarmente utile per rappresentare le logiche di controllo e le sequenze di azioni che devono essere eseguite da un PLC. Successivamente, abbiamo tradotto il diagramma SFC nel linguaggio a contatti, che rappresenta il programma come una serie di reti di contatti collegati in parallelo e in serie. La logica di controllo viene definita attraverso l'interconnessione dei contatti e delle bobine. Abbiamo poi definito e implementato gli schemi per la valutazione delle transizioni, il superamento delle transizioni e l'esecuzione delle azioni.

Attraverso la realizzazione di questo progetto, sono riuscito a capire come implementare un processo industriale a partire da zero. In particolare come sfruttare SFC per creare uno schema logico e quindi una prima rappresentazione del progetto. Questo progetto è stato molto utile anche a causa del grande impiego dei PLC nel settore industriale; Nonostante la semplicità del progetto rispetto ad un vero impianto industriale, mi ha fornito degli strumenti basilari realmente utili nel mondo del lavoro.

4 Controlli Automatici Progetto: Controllo Pendolo inverso

4.1 Introduzione

Il pendolo inverso è un sistema di profondo interesse per lo studio dei sistemi dinamici. In particolare consiste in un asta collegata ad un carrello, libero di muoversi lungo l'asse orizzontale. L'obiettivo del controllore che abbiamo implementato è quello di sfruttare il movimento laterale del carrello per mantenere l'asta in equilibrio perpendicolarmente al terreno.

4.2 Discussione progetto

Abbiamo deciso innanzi tutto di realizzare il pendolo fisicamente anziché in ambienete Simulink. Per fare ciò ci siamo serviti del meccanismo di movimento della testina di una stampante, del motore della stessa, e di un Arduino. Abbiamo innanzitutto ricavato un modello semplificato del sistema e successivamente linearizzato tali equazioni intorno al punto di equilibrio. Abbiamo scelto come punto di equibilibrio 180°, ipotizzando che l'asta diretta verso il basso sia a 0°. Utilizzando la trasformata di Laplace, siamo riusciti a ottenere le funzioni di trasferimento del sistema dalle equazioni linearizzate. Abbiamo utilizzato Matlab per analizzare la funzione di trasferimento ingresso-uscita del sistema che

abbiamo poi trasformato da tempo continuo al dominio di Tustin, per sintetizzare più semplicemente il controllore. Sfruttando i diagrammi di Bode ed il luogo delle radici, siamo stati in grado di ottenere la funzione di trasferimento del controllore che soddisfacesse i requisiti di controllo, per poi anti-trasformare dal dominio di Tustin a tempo discreto.

Una volta fatto ciò, tramite `ss` in Matlab abbiamo ottenuto le matrici del sistema nello spazio di stato (A, B, C, D). Queste matrici sono state utilizzate per implementare il controllore su Arduino, realizzando un sistema il quale ingresso era l'angolo dell'asta e la quale uscita corrispondeva al modulo del momento da applicare al motore per stabilizzare l'asta in posizione verticale. Nonostante il modello originale si basi su due ingressi (posizione del carrello, e angolo dell'asta), siamo comunque riusciti a mantenere l'asta in posizione verticale per qualche secondo, prima che il controllore fallisse.

Questo proggetto è stato il più importante svolto durante l'anno. Infatti oltre ad aver costruito un progetto a partire da zero, la curva di apprendimento è stata notevole. Infatti ne io ne il mio collega avevamo mai usato Arudino prima, perciò nonostante il progetto fosse molto difficile di per sé, molto tempo lo abbiamo passato a capire come usare Arduino. Inoltre sia il pendolo stesso, realizzato con materiali di ricico, che il sensore per l'angolo dell'asta (un potenziometro di scarsa qualità), hanno reso il sistema altamente soggetto a disturbi, e quindi poco robusto. Nonostante questo siamo riusciti a trovare un controllo che per alcuni secondi stabilizzasse il pendolo. Oltre quindi ad avere imparato ad usare Matlab, componenti elettronici, Arduino e ad aver applicato la teoria di Controlli Automatici, questo progetto ci ha anche regalato grandi soddisfazioni.

5 Ingegneria Internet e Web Progetto: Trasferimento file su UDP

5.1 Introduzione

Lo scopo di questo progetto è quello di progettare ed implementare in linguaggio C un'applicazione client-server su base UDP che permetta:

- Connessione client-server senza autenticazione;
- La visualizzazione sul client dei file disponibili sul server (comando list);
- Il download di un file dal server (comando get);
- L'upload di un file sul server (comando put);
- Il trasferimento file in modo affidabile.

La comunicazione tra client e server deve avvenire tramite un opportuno protocollo. Il protocollo di comunicazione deve prevedere lo scambio di due tipi di messaggi:

- Messaggi di comando: Sono i comandi che il client invia al server per richiedere l'esecuzione di una azione;
- Messaggi di risposta: Sono i comandi che il server in via al client in risposta ad un comando, come ad esempio l'esito di un' operazione.

5.2 Discussione progetto

L'architettura del sistema prende come riferimento lo schema Go-Back-N, sul quale si basa anche TCP. Per realizzare le due applicazioni client e server, abbiamo implementato particolare funzioni generiche in modo che potessero essere utilizzate su entrambi gli applicativi. Ad esempio la funzione che invia i pacchetti è la stessa sia per il client che per il server. Questo ci ha permesso di semplificare di molto l'implementazione, poiché gran parte del codice è condiviso da entrambe le applicazioni. Inoltre entrambe le applicazioni sfruttano il MultiThreading e sono MultiProcesso, cosi da permettere la gestione parallela di più richieste.

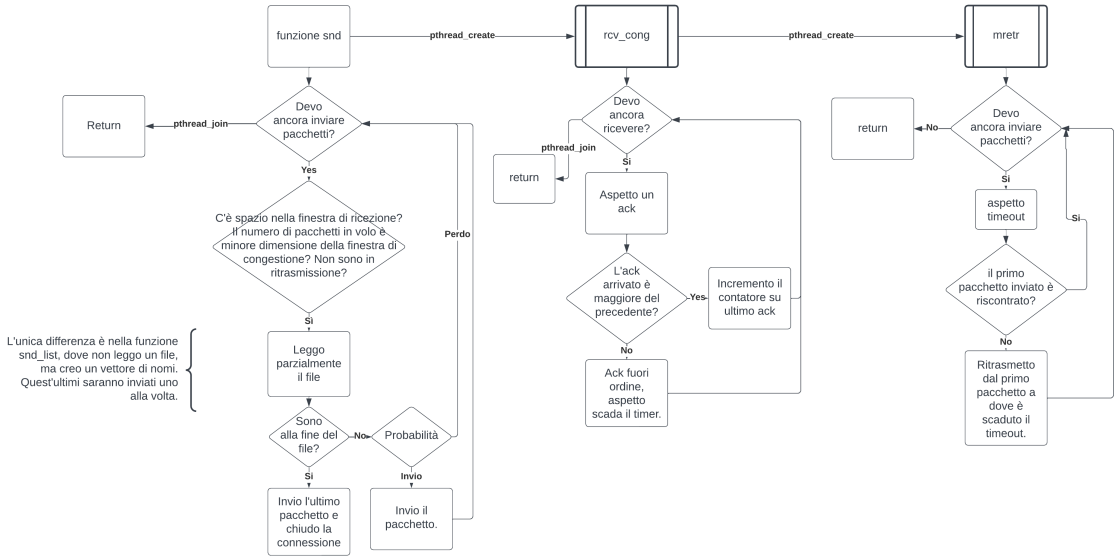

Figure 1: Schema invio messaggi

- Funzione snd: rappresenta tutte le funzioni di *Send* che ci permettono di inviare i pacchetti tra server-client e client-server.
- Rcv_cong: funzione che implementa la ricezione ed il controllo dei pacchetti.
- Mretr: realizza la ritrasmissione dei pacchetti in caso di perdita.

Una volta avviata l'applicazione server, il processo padre rimane in ascolto sulla socket principale, e per ogni richiesta di connessione, assegna una porta diversa a ogni processo child. Inoltre per evitare di realizzare una applicazione StopAndWait, è necessario ricevere e inviare messaggi contemporaneamente, rendendo di fatto imprescindibile l'uso di una architettura MultiThreaded. Per implementare la trasmissione affidabile, è stato necessario definire una struttura dati che rappresentasse un pacchetto. Nonostante quest'ultimo poi venga incapsulato in un datagramma UDP, è necessaria la struttura dati per permetterci di tenere traccia di alcune informazioni sul pacchetto stesso. In particolare abbiamo definito :

```
struct st_pkt{
    int id;
```

```
    int code;
    char pl[MAXLINE];
    int rwnd;
};
```

La variabile *id*, serve per numerare il pacchetto. Il *sender* quando invia un pacchetto tratterà *id* come se fosse il sequence number, quando invece lo riceverà tratterà *id* come se fosse l'ack number. Viceversa il *receiver*, in ricezione, tratterà *id* come il sequence number, e utilizzerà *id* come ack number quando lo trasmetterà.

La variabile *code*, viene utilizzata per definire le varie tipologie di pacchetto. Ad esempio $code = 0$ indica che il pacchetto è di tipo informativo; $code = 2$ indica che il pacchetto è usato per gestire le connessioni (come il caso di un pacchetto di richiesta di connessione da parte di un client). La stringa *pl* (payload) è un vettore di caratteri di grandezza predefinita *MAXLINE* che corrisponde a 4096 bytes. La variabile *rwnd* (receiver window), indica la dimensione della finestra di ricezione.

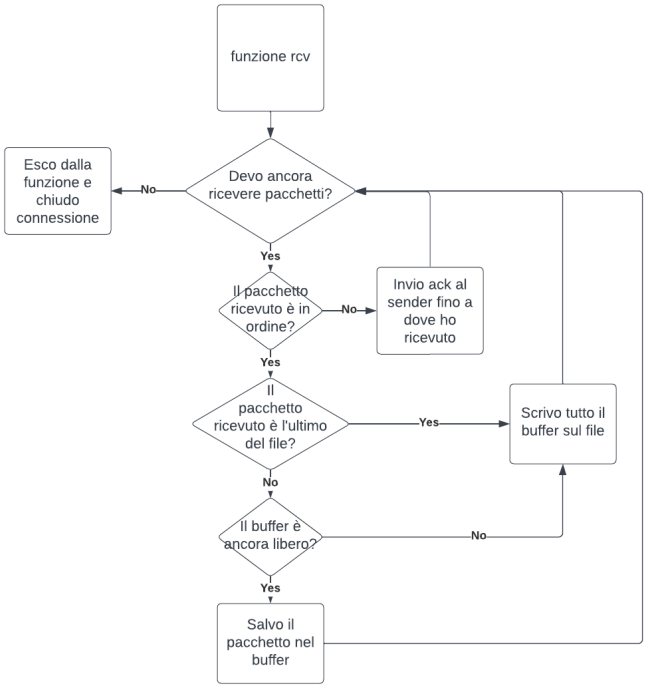

Figure 2: Schema ricezione messaggi

- Funzione rcv: rappresenta tutte le funzioni di ricezione che ci permettono di ricevere i pacchetti tra client-server e server-client.

Questo progetto è stato di grande aiuto soprattuto perchè ci siamo resi conto di quanto sia importante pianificare prima di iniziare un progetto. Infatti dopo qualche settimana di lavoro, molte idee che sembravano apparentemente funzionare, si sono rivelate incompatibili fra di loro, o impossibili

da implementare. Per questo motivo abbiamo ricominciato il progetto disegnando gli schemi di funzionamento prima di cominciare a scrivere codice. Per concludere, questo progetto ci ha permesso di mettere in pratica ciò che abbiamo imparato non solo nel corso in questione, ma anche dal corso di Sistemi Operativi, e di Fondamenti di Telecomunicazioni.

6 Teoria dei sistemi : Compiti di tirocinio

6.1 Introduzione

Durante il semestre di lezione, sono stati assegnati diversi compiti di tirocinio, in particolare io ho svolto il primo, il terzo ed il quarto.

6.1.1 Tirocinio 1

Lo scopo di questo primo tirocinio è quello di creare una o più funzioni su MATLAB in grado di fornire per un determinato sistema rappresentato nello spazio di stato, un controllo a tempo discreto che permetta di raggiungere uno stato desiderato in ν passi, minimizzando inoltre l'indice di costo: $J = \sum_{h=0}^{\nu-1} u^{'}(h)u(h)$. Per prima cosa mi sono assicurato che il sistema fosse raggiungibile, e successivamente ho calcolato il controllo ottimo $u(h)$. In una seconda funzione MATLAB ho creato un ciclo `for` che mi permettesse di calcolare l'ingresso ottimo per raggiungere lo stato desiderato in vari passi, in particolare da 2 ad ν_{max}. Tramite una terza funzione ho poi graficato l'andamento dell'indice di costo e delle traiettorie dello stato e dell'ingresso all'aumentare dei passi, di fatto osservando come l'indice di costo e l'intensità del controllo diminuissero all'aumentare dei passi. Ho poi ripetuto la procedura per alcuni casi particolari, come ad esempio per autovalori instabili o vicini all'instabilità.

6.1.2 Tirocinio 3

Lo scopo di questo tirocinio è quello di progettare un'osservatore dallo stato che permetta di stimare lo stato di un sistema massa-molla-smorzatore, e di testare in determinate situazione la sua efficenza. In particolare il sistema si compone di 2 masse che approssimano 2 carelli, collegati tra loro tramite una molla ed uno smorzatore, aventi inoltre attrito nullo con la superficie su cui si muovono.

La prima richiesta consisteva nel capire per quali valori dei parametri del sistema, ovvero in particolare le masse e i coefficenti di smorzamento e di elasticità, il sistema avessi autovalori reali e distinti, immaginari puri, e complessi coniugati. Per realizzare questi 3 diversi casi, basta agire sul coefficente di smorzamento. In particolare per $c = 0$ si hanno autovalori immaginari puri, per $c = 1$ complessi coniugati, mentre per $c > 1$ autovalori reali e distinti. Inoltre il sistema a causa della mancanza di attrito ha di per sè 2 poli nell'origine.

A questo punto per ognuno dei 3 casi ho realizzato 3 osservatori asintotici dello stato. Ho quindi tramite la formula di Ackermann trovato la matrice F tale che $V = -F'$ e per la quale la matrice $(A - VC)$ abbia tutti i poli nel semipiano complesso sinistro. In particolare al primo osservatore ho assegnato autovalori lenti tale che $Pdes(A) = (A + I)^4$, ovvero $\sigma(A - VC) = -1$. Al secondo osservatore ho assegnato autovalori veloci tale che $Pdes(A) = (A+10*I)^4$, ovvero $\sigma(A-VC) = -10$. Cosi ho realizzato gli osservatori con diverse velocità di convergenza e tale che la retroazione dallo stato sia anche asintoticamente stabile. Il terzo osservatore invece è stato progettato tale che la matrice V fosse il guadagno di un filtro di Kalman. Una volta terminato il progetto e inseriti i vari sistemi su Simulink ho verificato come lo stato stimato converga allo stato del sistema in tempo

finito, notando come l'osservatore lento converga più lentamente, quello veloce ha più *peaking* ma transitori più brevi, e quello di Kalman invece sia la via di mezzo ideale fra i due.

Successivamente ho studiato il comportamento del sistema in reazione a disturbi sulla misura di y, a distrubi in ingresso e rispetto a piccole variazioni parametriche del sistema. Ho osservato come l'osservatore di Kalman sia effettivamente il migliore in tutti i casi in termini di filtraggio dei disturbi e tempi di assestamento. L'osservatore veloce invece tende a filtrare male i disturbi, specialmente quelli molto veloci, ma tende a mantenere il tempo di assestamento piccolo. L'osservatore lento invece si comporta bene in termini di filtraggio dei disturbi, ma a discapito del tempo di asssestamento.

6.1.3 Tirocinio 4

Lo scopo di questo tirocinio è progettare un compensatore in retroazione dinamica dall'uscita che ottenga stabilità asintotica del sistema a ciclo chiuso e che garantisca errore di inseguimento nullo per eventuali riferimenti scelti. Per fare questo ho dovuto sfruttare l'osservatore progettato nel tirocinio 3. In particolare per prima cosa era richiesto di verificare che il sistema assicurasse già per costruzione, errore di inseguimento nullo per riferimenti a rampa. Ho verificato questa condizione non solo matematicamente, notando come i 2 poli in 0 del sistema fossero condizione necessaria e sufficente, ma anche tramite simulazione su Simulink.

Successivamente ho progettato 2 compensatori C_s in grado di stabilizzare e garantire errore di inseguimento nullo per riferimenti sinusoidali. Il primo con una dinamica di convergenza a 0 dell'errore più veloce rispetto al secondo.

Infine ho verificato come per piccole variazioni parametriche il sistema rimanesse asintoticamente stabile, e l'errore di inseguimento convergesse a 0.

Questi 3 tirocini mi sono stati molto utili non solo per mettere in pratica la teoria studiata durante il corso, ma anche per imparare ad usare gli ambienti MATLAB e Simulink. Sono riuscito infatti a trovare un significato pratico al problema del controllo ottimo, e all'osservatore dallo stato. Infatti l'applicazione ad un caso reale mi ha permesso di immaginare meglio il comportamento del sistema, e anche di poter riconoscere più in fretta il significato dei vari stati e delle uscite.