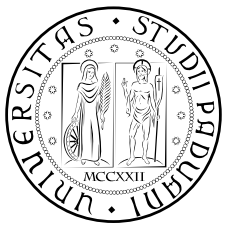

UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI TECNICA E GESTIONE DEI SISTEMI INDUSTRIALI
CORSO DI LAUREA TRIENNALE IN INGEGNERIA MECCANICA E MECCATRONICA

TESI DI LAUREA TRIENNALE

Studio e implementazione di un sistema assisted drive per robot manipolatori.

Relatore: Ch.mo Prof. Stefano Ghidoni

Laureando: Mattia Milanese
1051597-IMM

ANNO ACCADEMICO: 2016-17

La semplicità è l'ultima sofisticazione.
- Leonardo da Vinci

INDICE

ELENCO DELLE FIGURE

1 INTRODUZIONE

Esistono in letteratura e in ambito industriale diverse architetture per il controllo di un robot manipolatore ed hanno come obiettivo quello di fornire un segnale di comando agli attuatori dei giunti per far sì che il manipolatore esegua il task assegnato in modo corretto.
Tuttavia gli obbiettivi assegnati al robot possono rendere molto diversa la realizzazione di un controllo: nel caso più semplice il manipolatore è programmato per spostare l'organo terminale del robot, detto end effector, da un punto ad un altro attraverso una traiettoria nello spazio pre programmata prima dell'avvio del robot, senza avere un'interazione con l'ambiente esterno; in situazioni più complesse ad esempio si dovrà eseguire un controllo continuo di traiettoria rispettando dei vincoli sul percorso di posizione, velocità, accelerazione e con specifiche di precisione lungo la traiettoria stessa.
Inoltre anche la struttura meccanica può rendere diversa l'architettura di controllo: ad esempio uno schema valido per un manipolatore cartesiano non può essere utilizzato per un manipolatore antropomorfo; allo stesso modo un robot progettato con una struttura tale da compensare in modo automatico la componente gravitazionale agente sui bracci ha un controllo diverso rispetto a quello di un robot non equilibrato.
Queste diverse esigenze hanno portato il mondo dell'industria a sviluppare diverse tecniche più o meno sofisticate tra cui il controllo di forza.

1.1 CONTROLLO DI FORZA DI UN MANIPOLATORE

Al giorno d'oggi la maggior parte dei manipolatori industriali è programmata utilizzando un controllo di movimento il quale permette di seguire una traiettoria precisa dello spazio; a volte viene utilizzato un sistema di visione, che rende capace il robot di modificare il suo movimento in funzione dell'ambiente circostante; ma per alcune applicazioni in cui si ha un'interazione tra il robot e l'ambiente può essere più utile ottenere un feedback della forza applicata all'end effector per comandare in modo corretto il manipolatore.
Infatti il controllo del movimento puro a volte risulta essere inadeguato a lavorare con sistemi che devono percepire il contatto fisico con l'esterno perché incapace di limitare errori dovuti ad un comportamento instabile durante la lavorazione; inoltre può essere utilizzato solamente se il task viene pianificato accuratamente, cioè si ha una

conoscenza precisa del modello del manipolatore e dell'ambiente circostante. Quindi in molte situazioni si rende necessario affiancargli un controllo delle forze [2] di interazione che nascono tra l'end effector e le superfici esterne in modo da raggiungere un comportamento robusto.

1.1.1 *Schema generale di un controllo di forza*

Come affermato da Daniel E. Whitney in [11] per capire del tutto il funzionamento di un'architettura basata sul controllo di forza può essere utile ripercorrere i passi che portano alla progettazione di tale controllo:

- *Descrizione del Task*. Consiste nella descrizione di un modello che possa predire quali movimenti, posizioni, deformazioni e di conseguenza forze nasceranno a causa di una determinata interazione.
- *Relazioni movimento-forza*. Si tratta di analizzare quali movimenti (forze) nasceranno in risposta a possibili forze (movimenti), valutando se queste siano desiderabili oppure no, quali nascono a causa di interazioni passive, cioè dovute alla deformazione elastica dei corpi, e quali a causa di interazioni attive, dovute quindi al fatto che vengano alimentati gli attuatori.
- *Strategia di esecuzione del task*. Bisogna elaborare una strategia sulla base dei risultati trovati nei primi due punti con l'obbiettivo di raggiungere forze, movimenti, posizioni o velocità desiderate evitando invece i stati indesiderati.
- *Comandi logici*. Consiste nel scegliere i comandi di forza e velocità o una combinazione di entrambe per creare una certa risposta di forze e movimenti.
- *Controllo*. Il progettista deve implementare la sua strategia misurando le forze e i movimenti dell'end effector del robot per generare dei nuovi comandi in modo da avere in uscita le forze o i movimenti desiderati.
- *Stabilità*. Il controllo ovviamente crea degli anelli di retroazione che possono portare a problemi di instabilità che bisogna eliminare.

Da questi passi possiamo inoltre ricavare uno schema generale del controllo di forza valido per la maggior parte di architetture presenti oggi (Figura 1) e introdotta da [11].

Da questo si vede che in base al movimento che deve effettuare il robot vengono generati dei comandi per gli attuatori di ogni giunto. Questi vengono confrontati con i nuovi comandi provenienti dalla

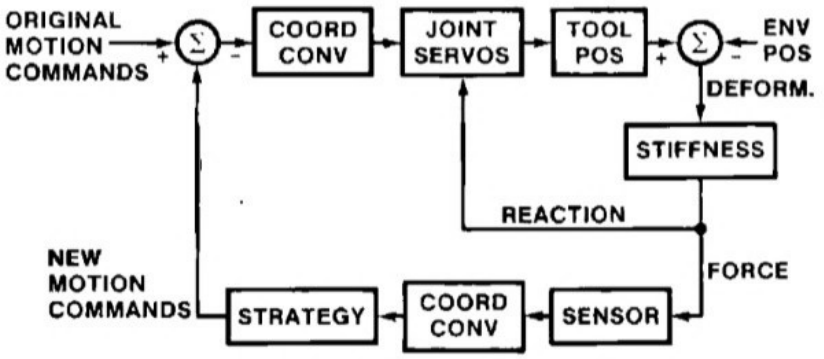

Figura 1: Schema generale controllo di forza

retroazione e in seguito convertiti in coordinate per gli attuatori del robot. Ad un certo punto quando avviene il contatto tra l'end effector e l'ambiente nascono delle forze di interazione, le quali reagiscono direttamente con i giunti del manipolatore modificandone il comportamento. Queste sono ricavate a partire dalla misura della variazione di posizione dell'end effector, e dal valore di rigidezza dell'oggetto con cui interagiscono. In base al valore di forza e alla strategia di controllo elaborata vengono generati dei nuovi comandi che saranno confrontati con quelli originali.

1.2 STATO DELL'ARTE: I TIPI DI CONTROLLO DI FORZA

Dall'idea di base di controllo di forza discussa nel paragrafo precedente sono nate in letteratura diverse architetture per far fronte ai diversi compiti che un robot manipolatore dovrà eseguire in ambito industriale. Queste si possono dividere in *controllo diretto* e *controllo indiretto*. La differenza sostanziale tra le due soluzioni è che mentre nel primo caso è necessario un esplicito anello di retroazione della forza, nel secondo quest'ultima viene ricavata a partire da altre relazioni. Tra i controlli indiretti di forza avremo ad esempio il controllo di rigidezza e controllo di impedenza poiché prevedono una retroazione di posizione e/o velocità. Tra quelli diretti troviamo invece il controllo Ibrido di forza e velocità, e il controllo di forza puro.

1.2.1 *Controllo di forza puro*

Il controllo di forza puro [10] come detto è un controllo diretto, prevede quindi la misurazione attraverso un sensore della forza esterna e il confronto di questa con la forza di riferimento da applicare. Il tutto avviene attraverso un anello di retroazione negativa e con l'utilizzo di un regolatore proporzionale e derivativo [4] per stabilizzare il sistema. Con questo tipo di controllo si presentano però a livello pratico dei problemi: il rumore presente durante l'acquisizione dei valori di forza non permette di implementare l'azione derivativa. Questa infatti alle alte frequenza ha un guadagno infinito il quale va ad amplificare il rumore compromettendo quindi la bontà della misurazione. Come conseguenza un tipico sistema di controllo di forza non si basa soltanto sulla misurazione della forza, ma anche sulla misurazione della velocità oppure della posizione.

1.2.2 *Controllo di rigidezza*

Il controllo di rigidezza (*Stiffness control*) [7] è utilizzato nelle applicazioni dove viene richiesto di assegnare un ben preciso valore di forza che l'end effector deve applicare su una superficie esterna in condizioni stazionarie. È un controllo indiretto quindi la forza desiderata può essere ottenuta in due modi:

- Utilizzando una retroazione di posizione. In questo caso le forze vengono calcolate rappresentando l'interazione tra il manipolatore e l'ambiente esterno come un sistema formato da una molla equivalente con un determinato coefficiente di rigidezza.
- Utilizzando una retroazione di velocità. In questo caso il manipolatore e l'ambiente esterno sono rappresentati come un sistema formato da uno smorzatore equivalente.

Analizziamo come sia possibile arrivare a definire uno schema di controllo. Per semplicità consideriamo il caso ad un grado di libertà, quindi studiamo l'equilibrio di forze lungo un determinato asse. Tutto questo sarà utile in seguito per capire dove si inserirà l'esperienza che verrà proposta. Consideriamo quindi l'interazione tra end effector e l'ambiente esterno (Figura 2).
possiamo definire l'equazione:

$$f_e = K(x_e - x_r) \tag{1}$$

la quale esprime la forze esterna f_e applicata in base alla variazione della posizione della punta del manipolatore durante l'interazione con la superficie esterna e al coefficiente di rigidezza della superficie stessa. Ritornando alla Fig.2 possiamo quindi definire x_r come la

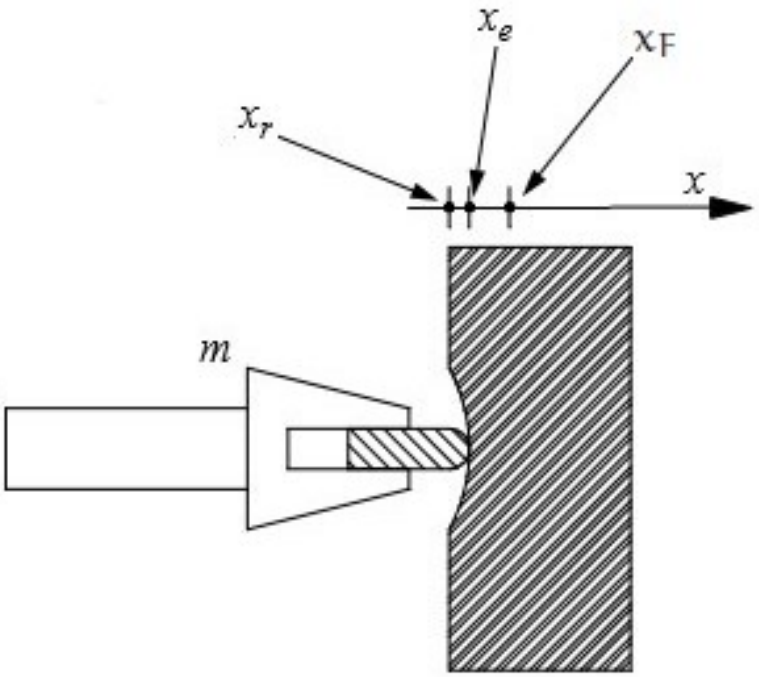

Figura 2: Controllo di rigidezza, calcolo della forza esterna

posizione di 'riposo' dell'end effector, x_e la posizione effettiva e K il coefficiente di rigidezza dell'ambiente esterno, mentre x_F è la posizione di riferimento dove deve arrivare la punta del manipolatore. Consideriamo ora l'equazione dinamica del sistema robot e ambiente:

$$m\ddot{x} + f_e = f_c \,, \tag{2}$$

dove f_c è la forza di comando assegnata, cioè la forza che vogliamo si applicata dal manipolatore. La legge di controllo di rigidezza proposta da [8] per f_c è di tipo proporzionale-derivativo:

$$f_c = K_p(x_F - x_e) + K_d(\dot{x_F} - \dot{x_e}) \,, \tag{3}$$

dove Kp indica il guadagno proporzionale, K_d il guadagno derivativo, mentre $\dot{x_F}$ e $\dot{x_e}$ sono le velocità rispettivamente dei punti x_F e x_e. Inserendo in quest'ultima equazione (3) quella precedente (2) dopo una serie di passaggi troviamo l'espressione utilizzata per ricavare il sistema di controllo rappresentato in Figura 3:

$$m\ddot{x_e} + K_d\dot{x_e} + (K - Kp)x_e - Kx_r = K_p C_F(f_d - f_e) \,, \tag{4}$$

dove troviamo definiti oltre ai parametri già introdotti precedentemente l'accelerazione $\ddot{x_e}$, l'errore di forza f_d, e il coefficiente C_F. Pos-

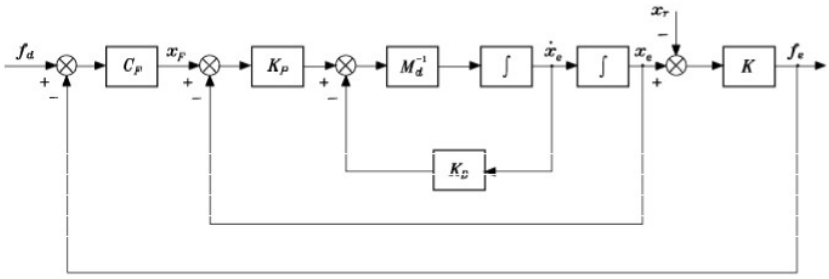

Figura 3: Schema del controllo di rigidezza

siamo fare ora alcune considerazioni su alcuni parametri di questo schema di controllo:

- C_F rappresenta un altro controllo che viene effettuato. Ha una componente proporzionale K_F e una integrativa K_i:

$$C_F = K_F + K_i \int^t () dx \,. \tag{5}$$

  Permette inoltre di esprimere una relazione tra x_F e l'errore di forza:

$$x_F = C_F(f_d - f_e) \,. \tag{6}$$

- Il guadagno di posizione K_p può essere interpretato come la rigidezza desiderata del sistema manipolatore e ambiente esterno. L'algoritmo di controllo proposto viene ricordato come "controllo di rigidezza" proprio perché il valore desiderato di forza viene assegnato modificando opportunamente il valore di rigidezza equivalente del sistema.
- Il guadagno di velocità K_d invece può essere rappresentato come il coefficiente di smorzamento del sistema, consente di variare la velocità con cui viene raggiunta la stazionarietà. Una variante del controllo di rigidezza è il cosiddetto *damping control* (Figura 4) che viene realizzato essenzialmente utilizzando solamente una retroazione di velocità assegnando il valore di smorzamento desiderato per ottenere la forza richiesta.

Possiamo concludere che in un controllo di rigidezza rappresentato nella Figura 3, per assicurare il perfetto funzionamento del sistema e la sua stabilità, bisogna impostare i valori di K_F, K_i, K_e e K_d.

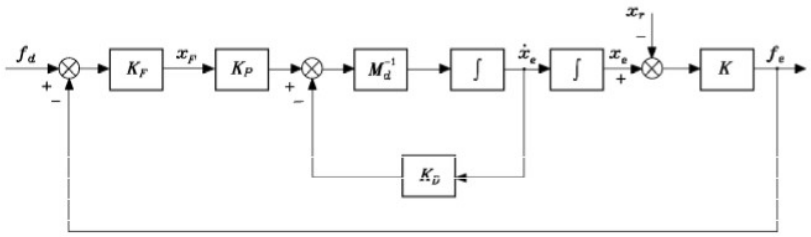

Figura 4: Schema controllo di forza con la sola retroazione della velocità

1.2.3 *Controllo di impedenza*

Se siamo invece in una situazione dove non vi è il bisogno di regolare la forza in modo accurato ma soltanto di garantire che rimanga all'interno di determinati limiti viene preferito l'utilizzo di un controllo di impedenza (*impendance control* [6]). In questa situazione l'interazione tra manipolatore e ambiente viene rappresentata da un sistema equivalente massa-molla-smorzatore il quale viene descritto da un insieme di equazioni differenziali di secondo ordine che corrispondono all'impedenza dinamica del sistema complessivo. Infatti facendo riferimento all'analogia elettro-meccanica che associa alle forze le tensioni e alla velocità la corrente, il rapporto tra forze e velocità in un sistema meccanico può essere visto come un'impedenza meccanica del sistema stesso, definita nel dominio di Laplace come:

$$Z(s) = \frac{F(s)}{v(s)}, \tag{7}$$

dove $F(s)$ e $v(s)$ rappresentano rispettivamente la trasformata della forza applicate e della velocità dell'end effector. L'obiettivo di questo controllo è quello di scegliere opportuni coefficienti di rigidezza e smorzamento del modello di impedenza per ottenere un buon compromesso tra l'accuratezza del posizionamento e l'entità delle forze applicate.

1.2.4 *Controllo di forza ibrido*

Un'ultima architettura di controllo molto utilizzata al giorno d'oggi è il controllo Ibrido di posizione e forza [5] che come suggerisce il nome permette di regolare sia il movimento del robot che l'interazione che esso ha con l'esterno. Permette quindi di seguire determinate traiettorie e contemporaneamente di assegnare dei valori precisi di forza anche in direzioni differenti rispetto a quelle del moto. È sicuramente la soluzione più complessa da implementare, ma anche quella che offre possibilità di utilizzo più ampie per eseguire compiti più complicati in modo ottimale.

1.3 APPLICAZIONI DEL CONTROLLO DI FORZA

Per capire meglio i vantaggi che può offrire un controllo di forza possiamo analizzare alcune delle applicazioni in cui viene utilizzato oggi:

- *Telemanipolazione*. In un sistema 'master slave' un operatore umano attraverso una periferica di controllo, comanda da remoto un robot 'schiavo' riuscendo ad esempio a manipolare oggetti con precisione e facilità.
- *Sistemi multi-robot*. In presenza di un sistema composto da due o più robot manipolatori che collaborano alla lavorazione e movimentazione dello stesso oggetto, il controllo di forza riesce ad eliminare eventuali piccoli errori di posizione evitando che l'oggetto possa sfuggire dalla presa dei manipolatori o al contrario sia sottoposto a pressioni troppo elevate.
- *Mani Robotiche.* Un feedback di forza è essenziale affinché una mano robotica riesca a svolgere un qualsiasi compito di manipolazione.
- *Riconoscimento di superfici.* Permette alla punta del robot di seguire la superficie di un oggetto sconosciuto, utilizzando le informazioni di forza ottenute per modificare gli input di posizione e velocità.

1.3.1 *Assisted drive system*

Un sistema Assisted-Drive è un pacchetto di software sviluppati per introdurre delle funzionalità che migliorino la guida manuale del robot manipolatore. Queste hanno tutte l'obiettivo di facilitare il lavoro dell'operatore umano il quale deve poter spostare il robot nella posizione desiderata interagendo con esso attraverso una mano, senza incontrare resistenza ma al contrario deve essere il manipolatore a seguire il movimento imposto dall'operatore. Se analizziamo il funzionamento logico di un sistema Assisted Drive possiamo distinguere i seguenti passi:

- Il robot è inizialmente fermo e l'operatore effettua una pressione sull'end effector.
- Vi è un'interazione tra l'ambiente esterno e il manipolatore che causa una modifica della posizione e velocità di quest'ultimo.
- Viene registrata una forza dai sensori presenti sull'end effector del robot.

- Una strategia di controllo che tiene conto della forza esterna e della sua direzione (attraverso i riferimenti di posizione e velocità) elaborerà i comandi per far sì che il manipolatore segua il movimento imposto dall'operatore.

Possiamo facilmente trovare una corrispondenza tra i punti appena descritti e il schema presentato nel Paragrafo 1.1.1, si può quindi affermare che un sistema Assisted Drive è sostanzialmente basato su un controllo di forza del manipolatore.

1.3.2 *Stato dell'arte*

I sistemi Assisted drive oggi non sono ancora molto utilizzati all'interno del mondo industriale perché gli viene ancora preferito l'utilizzo della modalità Free Drive [1] presente in quasi tutti i robot manipolatori. Infatti anche quest'ultimo permette di muovere il manipolatore applicando una forza con la mano in modo da dare al robot la posizione che vogliamo, però a differenza della modalità Assisted Drive richiede di applicare una forza maggiore da parte dell'operatore per gli attriti presenti, affinché inizi ad esserci un movimento assistito. Viene ancora utilizzata per la sua facilità di realizzazione: non viene dato nessun comando agli attuatori e il robot viene lasciato libero di muoversi e quindi di esser trascinato nella posizione desiderata, non deve esser presente alcun sensore di forza né deve essere sviluppato alcun algoritmo per il controllo del movimento del manipolatore. Questi vantaggi però portano a livello pratico a dei risultati peggiori soprattutto per quanto riguarda la facilità di utilizzo e la precisione del posizionamento del braccio robotico. Possiamo renderci conto di questo analizzando una delle soluzioni migliori del mercato proposta dalla Robotiq [9] la quale ha introdotto nel software per la gestione dei robot manipolatori un sistema Assisted Drive che ha diverse funzionalità per la guida manuale:

- Movimentazione lungo i 6 assi, simile alla modalità free drive.
- Movimentazione scara, permette la traslazione nei piani X,Y,Z e la rotazione dell'end effector solo lungo l'asse z.
- Movimentazione cartesiana, permette solo la traslazione lineare lungo i tre assi.
- Movimentazione piana, che mantiene l'end effector sempre nello stesso piano.
- Orientazione automatica dell'end effector in una delle direzioni degli assi.
- Regolazione della velocità del movimento.

Abbiamo quindi non solo la possibilità di guidare il robot nella posizione desiderata ma anche vincolare i movimenti che il robot può effettuare, ad esempio abilitando la movimentazione piana potremmo impedire al robot manipolatore di muoversi verticalmente permettendogli di spostare l'end effector solo in dei punti appartenenti al piano in cui stiamo lavorando.
Oggi i manipolatori con software per l'Assisted Drive vengono utilizzati in applicazioni che vedono la stretta collaborazione tra robot e operatore come ad esempio nell'ambito odontoiatrico, oppure viene implementato per aiutare lo svolgimento di alcune attività secondarie come la manutenzione o insegnare manualmente al robot la traiettoria che l'end effector deve seguire. Nei capitoli successivi parleremo delgli esperimenti di laboratorio in cui cercheremo di implementare un sistema di assisted drive per il robot manipolatore UR5 (Paragrafo 2.2). Nella prima parte verranno spiegate le condizioni iniziali di lavoro e le ipotesi semplificative effettuate per descrive il fenomeno attraverso un modello matematico. Passeremo quindi ad analizzare attraverso Matlab e Simulink i dati ottenuti sperimentalmente, concludendo con l'implementazione in un nodo ROS del codice necessario per comandare il robot.

2

SVILUPPARE IL SISTEMA ASSISTED DRIVE

In questo capitolo si analizzerà l'obbiettivo di questa esperienza, in particolare parleremo del robot manipolatore utilizzato, dei dati che questo fornisce e che in seguito verranno elaborati, discutendo infine su alcune condizioni particolari di svolgimento.

2.1 FUNZIONAMENTO E OBIETTIVI DEL SISTEMA ASSISTED DRIVE

In questa esperienza di laboratorio cercheremo di implementare un semplice sistema di Assisted drive per il manipolatore UR5. Come discusso nei capitoli precedenti il robot dovrà esser capace di seguire la direzione imposta dalla nostra mano quindi di reagire all'applicazione di una pressione esterna con un movimento ad una determinata velocità la quale avrà lo stesso verso della forza applicata (Figura 6 a sinistra). Nel nostro caso non vedremo l'Assisted drive come un controllo di forza completo ma andremo soltanto ad analizzare l'anello di retroazione della velocità che come abbiamo visto precedentemente viene utilizzato per realizzare il *damping control*. L'obbiettivo sarà quindi quello di definire un modello che possa rappresentare l'interazione tra il robot manipolatore e l'ambiente esterno e di ricavare da questo una stima della velocità che deve avere la punta del robot.

2.2 IL ROBOT MANIPOLATORE

Il robot utilizzato durante l'esperienza è il manipolatore UR5 (Figura 5) prodotto dalla società Universal Robotic [9] e utilizzato nelle attività di laboratorio da studenti e dottorandi del IAS-Lab (*Inteligent Autonomous System Laboratory*). É formato da 6 giunti rotanti e ha una portata di carico utile fino a 5 kg con uno sbraccio massimo di 850 mm. Queste caratteristiche gli permettono di essere utilizzato a livello industriale in qualsiasi applicazione che richieda un processo automatizzato come ad esempio nella avvitatura, incollaggio e saldatura, prelievo e posizionamento ma anche assemblaggio e controllo qualità. Il robot è inoltre provvisto di un'interfaccia utente grafica la quale però non viene utilizzata dagli utenti del laboratorio. Al suo posto viene utilizzato Ros (*Robotic Operating System*) un framework con il quale avremo a che fare nel corso dell'esperienza. Esso fornisce diverse funzionalità utili per lo sviluppo di software di controllo del robot: al loro interno i framework possono includere programmi di sostegno, compilatori, librerie di codice, set di strumenti e API (*Appli-*

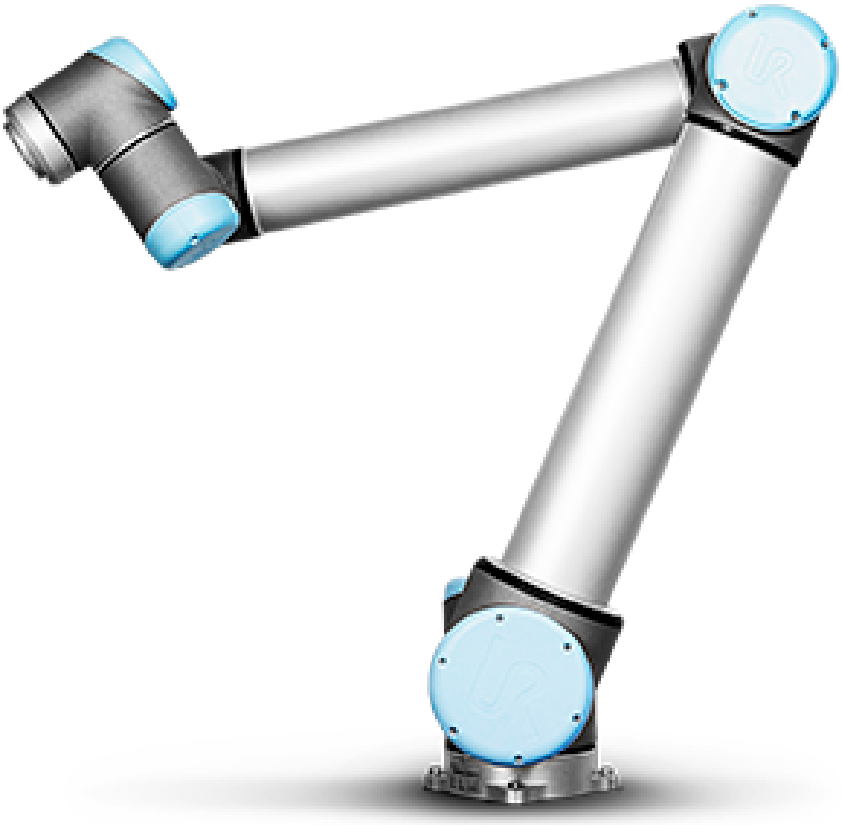

Figura 5: Il robot manipolatore UR5

cation programming interfaces), che insieme consento lo sviluppo di un progetto o di un sistema. Grazie a questi software è possibile ottenere e inviare da remoto alcune informazioni che possono essere utilizzate per il controllo del robot. Nel nostro caso vista l'attività che deve svolgere il robot, i dati che dobbiamo prelevare sono i valori di forza al TCP (*Tool center point*) del robot, mentre dovremo indicare quale accelerazione e quale velocità cartesiana dovrà avere l'end effector.

2.3 VINCOLI DI SVILUPPO

Per lo svolgimento dell'esperienza sono state considerate alcune ipotesi e poste alcune condizioni per semplificarne lo svolgimento le quali comunque permettono di ottenere allo stesso dei risultati accettabili:

- Per prima cosa l'esperienza verrà svolta considerando una sola direzione cartesiana lungo la quale possa essere esercitata una forza esterna e di conseguenza possa avvenire il movimento: l'asse z. I risultati ottenuti potranno essere in seguito replicati anche per gli altri assi.
- Le informazioni che devono essere inviate al manipolatore saranno un riferimento di velocità e uno di accelerazione del-

l'end effector. Il calcolo dei comandi da assegnare e il controllo per raggiungere tale velocità non viene preso in considerazione essendo già sviluppato all'interno del sistema operativo del robot.

- L'informazione di forza inviata dal manipolatore è data rispetto ad un sistema di riferimento relativo alla base del robot, mentre la velocità assegnata è espressa rispetto ad un sistema di riferimento solidale con l'end effector. Questo ci costringe a considerare la punta del robot orientata in modo tale da far coincidere i due sistemi di riferimento per evitare la situazione mostrata in Figura 6.

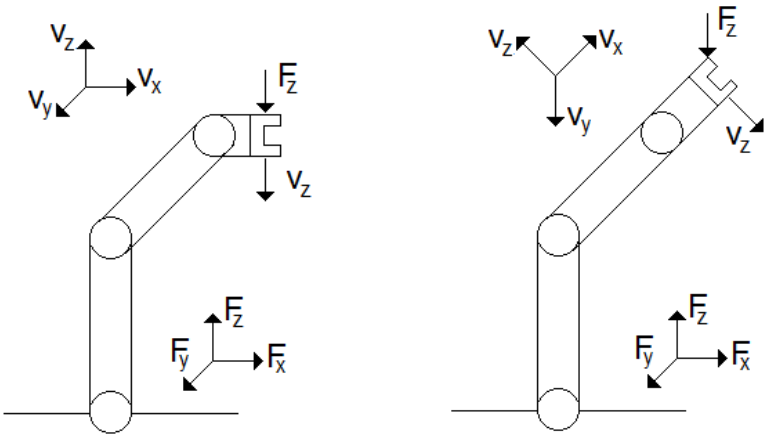

Figura 6: Relazione forza e velocità

- Per lo svolgimento dell'esperienza è necessario conoscere l'inerzia del robot e di conseguenza la sua massa. Purtroppo è impossibile ottenere questi valori mentre il manipolatore si trova in movimento, verrà di conseguenza fatta una stima in base alla geometria e ai materiali di cui è composto.

Utilizzando queste ipotesi sarà possibile nel capitolo successivo definire un modello del sistema, e svolgere delle simulazione per valutare il comportamento reale del robot. Viste le semplificazioni, i risultati ottenuti potrebbero non descrivere in modo corretto l'esperimento, ma allo stesso modo permettono di effettuare un'analisi ragionevole del sistema.

3

IL MODELLO MATEMATICO DEL SISTEMA

In questo paragrafo verrà introdotto il modello che descrive l'interazione tra il robot e l'ambiente esterno, il quale ci permetterà di effettuare una stima della velocità dell'end effector a partire dalla forza applicata. Ricaveremo quindi le equazioni nel dominio del tempo e nel dominio di Laplace.

3.1 CALCOLO DELLA FUNZIONE DI TRASFERIMENTO DEL SISTEMA

Per descrivere l'esperienza si è scelto inizialmente di rappresentare il robot come un corpo rigido di massa m, sottoposto ad una forza esterna $F(t)$, quindi in movimento con un'accelerazione $a(t)$ e velocità $v(t)$. Questa relazione dinamica è espressa dalla seconda legge di Newton:

$$F(t) = ma(t); \tag{8}$$

utilizzando questa è facile ricavare, con una serie di semplici passaggi, una formula che ci permetta di ottenere la velocità a partire dalla forza esterna impressa:

$$v(t) = v(0) + \frac{1}{m}\int F(t)dt\,, \tag{9}$$

dove $v(0)$, la velocità all'istante iniziale, può essere trascurata ipotizzando il robot inizialmente fermo. La (9) può essere riscritta nel dominio di Laplace:

$$V(s) = \frac{1}{m}\frac{1}{s}F(s)\,, \tag{10}$$

dove $V(s)$ e $F(s)$ sono le trasformate rispettivamente di $v(t)$ e $F(t)$, mentre l'integrale nel dominio di Laplace diventa $\frac{1}{s}$.

Ci si può facilmente rendere conto che questo modello per la stima della velocità anche se valido, risulta inadatto per descrivere il comportamento del robot, pertanto non può essere utilizzato in questo specifico caso. Infatti una caratteristica basilare che deve essere presente in un sistema assisted drive è la capacità di fermarsi quando dall'esterno viene applicata una forza nulla. Per raggiungere questo

obiettivo è necessario introdurre un termine smorzante, il quale porti ad un valore nullo l'uscita del sistema a regime. Si ha quindi che il sistema dinamico può esser descritto dalla seguente equazione:

$$F(t) = ma(t) - Bv(t)\,, \tag{11}$$

dove è stato introdotto il termine B, che rappresenta il coefficiente di smorzamento del sistema. Possiamo anche riscrivere la precedente relazione nel dominio di Laplace:

$$F(s) = msV(s) - BV(s)\,, \tag{12}$$

e ricavare la funzione di trasferimento che lega la velocità in uscita e la forza in ingresso:

$$W(s) = \frac{1}{Kd}\frac{1}{(1+\frac{m}{Kd}s)}\,; \tag{13}$$

mentre il sistema nel suo complesso è rappresentato dallo schema in Figura 7.

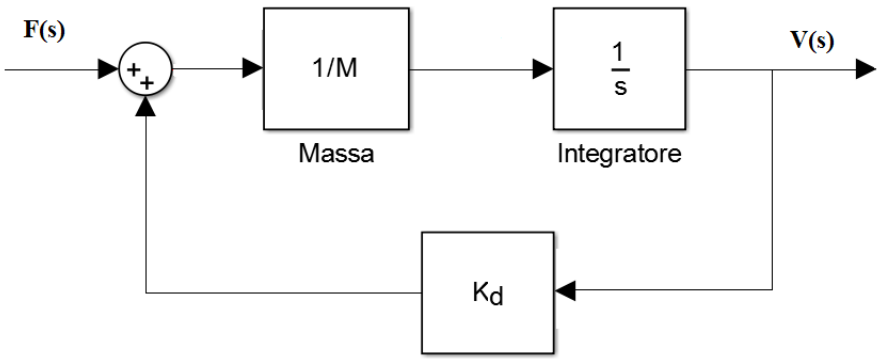

Figura 7: Modello per la stima della velocità del robot

La funzione di trasferimento (13) è formata da:

- un polo semplice $p_1 = -\frac{K_d}{m}$ con $\tau = \frac{m}{K_d}$.
- un guadagno di Bode $K_B = \frac{1}{K_d}$.

Dato che sia la massa m che il guadagno K_d sono due valori sempre positivi il sistema non presenta problemi di instabilità. Si può notare inoltre che in (13) e nella Figura 7 si è scelto di rappresentare il coefficiente di smorzamento B come un guadagno di velocità K_d. Infatti

analogamente a quanto succede nel *damping control*, durante l'esperienza uno degli obbiettivi sarà quello di regolare K_d in modo da ottenere un comportamento adeguato. Agendo su questo parametro infatti si può:

- Modificare la prontezza del robot. Trattandosi di un sistema elementare del primo ordine il segnale in uscita arriverà a regime più velocemente possibile quanto più piccola è la costante di tempo τ. Nel nostro caso visto che $\tau = \frac{m}{K_d}$ per migliorare il tempo di risposta del robot si dovrà aumentare il guadagno di velocità K_d.
- Regolare la velocità. Scegliendo un guadagno di velocità e di conseguenza un determinato coefficiente di viscosità, imponiamo al robot una ben precisa dinamica e di conseguenza la velocità desiderata.

3.2 SVILUPPO DEL MODELLO DEL ROBOT

In questi paragrafi verranno introdotte alcune funzionalità per migliorare il comportamento del modello precedentemente trovato, e per rendere l'uscita il più coerente possibile con il segnale di forza in ingresso. Tutto ciò verrà motivato in base alle simulazioni effettuate con Matlab e Simulink e ai risultati ottenuti eseguendo l'esperimento vero e proprio.

3.2.1 *Saturazione di accelerazione e velocità*

Per prima cosa si è deciso di inserire nel modello dei saturatori di velocità e accelerazione (Figura 8). Questa scelta è stata effettuata per

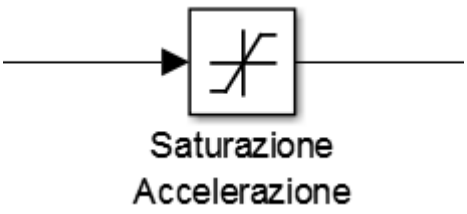

Figura 8: Blocco saturazione accelerazione

permetterci di lavorare in condizioni di sicurezza e per evitare che il robot abbia dei comportamenti ingestibili dall'utente, i quali possono anche portare al blocco automatico del robot stesso. Si è quindi deciso di limitare la velocità a dei valori compresi tra $-0,5\,m/s$ e $0,5\,m/s$ considerato che lo sbraccio massimo del manipolatore è di $0,85\,m$, mentre l'accelerazione è stata limitata a dei valori compresi tra $-1\,m/s^2$ e $1\,m/s^2$.

3.2.2 *Il guadagno di velocità*

Come già detto nel paragrafo 3.1 il guadagno di velocità è un parametro molto importante che deve essere fissato per imporre al sistema una certa dinamica. Il comportamento che vogliamo raggiunga il robot prevede due caratteristiche:

- Per quanto possibile il robot non deve mai superare i limiti di velocità.
- Il robot deve rispondere prontamente ad una sollecitazione esterna.

Per determinare un valore adeguato di K_d che soddisfi questi due requisiti, si è deciso di analizzare la risposta del sistema quando in ingresso viene dato un gradino con un modulo unitario. In base a questa prova (Figura 9) si è scelto un $K_d = 200$, in quanto permette di avere un tempo di salita $t_s = 0.22$ ms abbastanza basso e un valore della velocità a regime che rimane al di sotto del limite critico imposto precedentemente, anche se vengono applicate delle forze che superano 100 N.

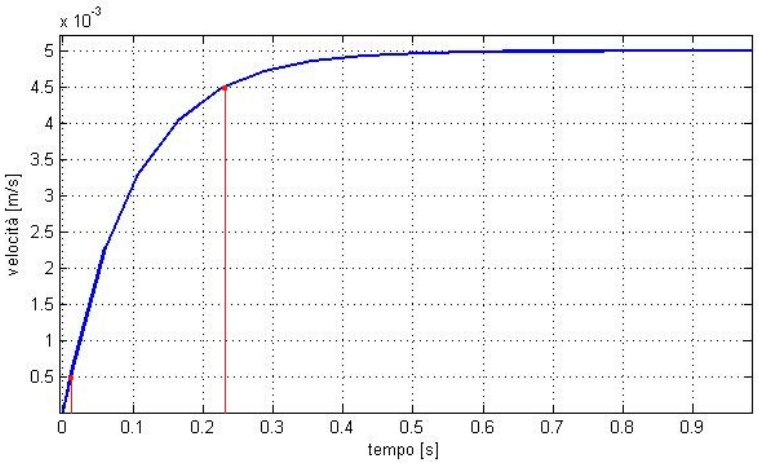

Figura 9: Risposta impulsiva del sistema

3.2.3 *Risposta del modello al segnale di forza*

Per effettuare la simulazione è stata effettuata una prima prova utilizzando il modello ricavato nel paragrafo precedente; sono stati inoltre raccolti dei valori di forza reali riferiti al TCP (*Tool Center Ponit*) del robot, per effettuare in contemporanea un'analisi del fenomeno attraverso Matlab e Simulink. Questi dati di forza vengono forniti da ROS come una serie temporale di campioni ad una frequenza di 125 Hz; pertanto sono stati interpolati per formare quello che sarà il segnale in ingresso al modello dinamico. Un esempio di tale segnale è rappresentato in Figura 10, ottenuto effettuando inizialmente una pressione

verso il basso (quindi secondo il sistema di riferimento definito nel capitolo 2 una forza negativa), in seguito una pressione verso l'alto, per finire nuovamente con una pressione verso il basso. I valori di

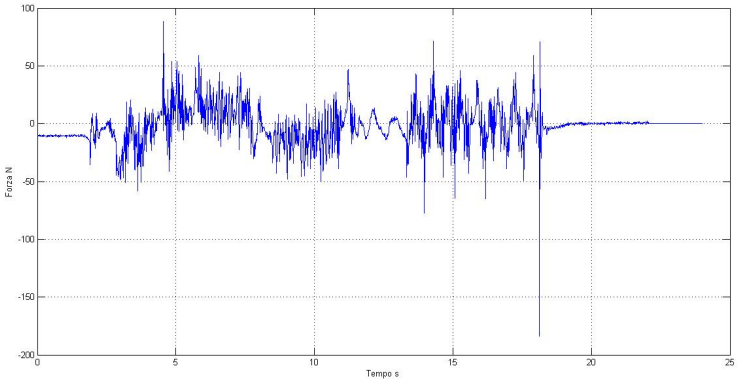

Figura 10: Un esempio del segnale di forza

velocità durante tale prova invece sono illustrati in Figura 11.
Da questa si può facilmente capire qual'è effettivamente stato il com-

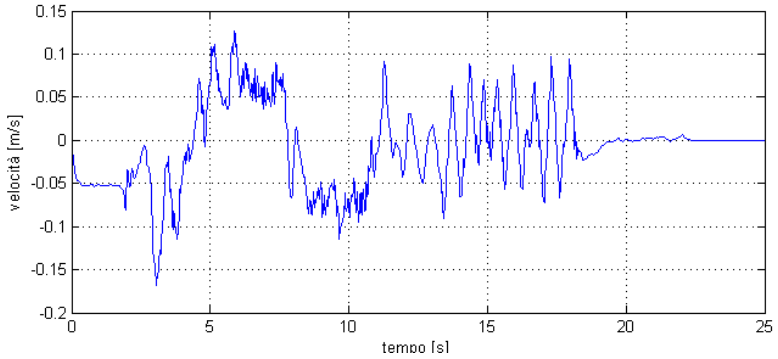

Figura 11: Velocità in uscita dal modello iniziale

portamento del robot: le velocità non corrispondo alle sollecitazioni effettuate sul manipolatore: questo si è mosso in modo errato non solo quando veniva applicata una forza, ma anche quando non è stato perturbato da un fattore esterno. Questo risultato però era prevedibile, infatti ritornando alla Figura 10 si può chiaramente notare che i valori di forza non sono molto precisi. Evidente è la presenza di rumore alle alte frequenze e di una componente continua nei primi secondi dell'interazione: questa situazione è dovuta al fatto che i dati non vengono raccolti da un sensore di forza, ma stimati dal sistema di controllo del robot a partire dalla corrente applicata agli attuatori dei giunti.

3.2.4 *Modello con l'annullamento della componente continua*

Sempre dalla figura 10 è possibile notare, durante gli istanti iniziali, la presenza di una componente continua della forza nonostante non venga esercitata alcuna sollecitazione nei confronti del robot. Questa ha causato alcuni problemi: tale forza in ingresso al modello del sistema ha portato ad un'accelerazione costante, con la conseguenza che il robot si è mosso anche senza la presenza di una sollecitazione. Per eliminare questo comportamento si è deciso di imporre a zero i valori di forza contenuti all'interno dell'intervallo $[-15\,\mathrm{N}, 15\,\mathrm{N}]$, agendo quindi anche su eventuali oscillazioni che si potrebbero presentare alla fine dell'interazione. Questa azione è stata implementata nel modello studiato utilizzando le proprietà dei comparatori: il segnale iniziale viene confrontato con un riferimento e se la relazione imposta viene rispettata, restituisce un valore booleano positivo. Per raggiungere il nostro obbiettivo sono stati utilizzati due comparatori in parallelo (Figura 12) così da formare un segnale con valore 1 quando $F < -15$ o $F > 15$, o quando $-15 < F < 15$. Otteniamo quindi un'onda quadra che moltiplicata per il segnale originale, permette di annullare i valori di forza all'interno dell'intervallo scelto, mantenendo inalterati i restanti. Utilizzando questa soluzione è stato raggiunto il risultato

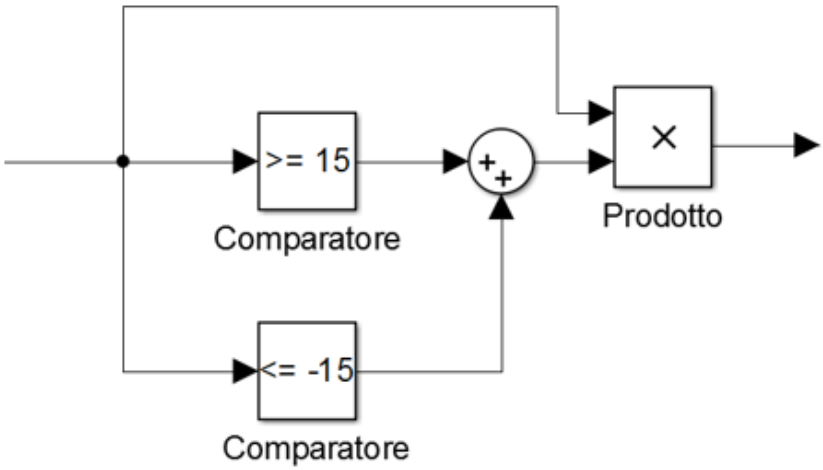

Figura 12: Schema per annullare la componente continua

sperato: dalla Figura 13 si può vedere che nel tratto iniziale e nelle zone dove la forza era minore dei limiti imposti, la velocità dell'end effector si è annullata. L'uscita e di conseguenza il comportamento del robot però non può ancora considerarsi soddisfacente: è presente ancora una grande quantità di rumore e l'andamento della velocità non rispecchia ancora le sollecitazioni esercitate sul robot.

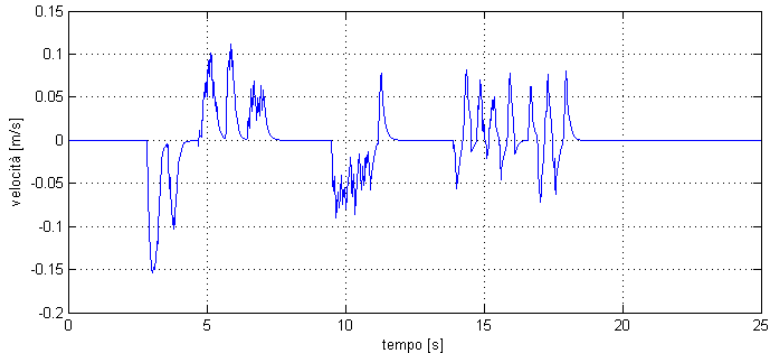

Figura 13: Velocità in uscita dal modello con l'annullamento della componente continua

3.2.5 *Modello con il filtro passa basso*

Prima di proseguire con un nuovo esperimento è stato deciso di progettare, sulla base dei dati già raccolti, un filtro passa basso con cui migliorare il comportamento dei valori di forza acquisiti. Per l'implementazione è stata effettuata attraverso Matlab un'analisi in frequenza del segnale (Figura 14), così da trovare la frequenza ottimale per attenuare nel miglior modo possibile il rumore. Si è deciso di utilizza-

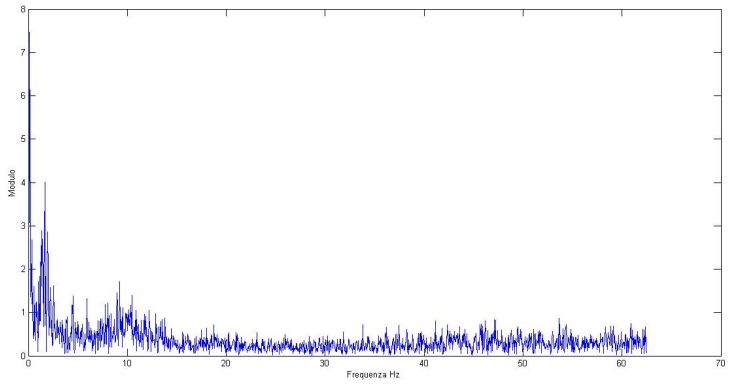

Figura 14: Spettro del segnale di forza

re un filtro passa basso e di conseguenza di tagliare tutte le frequenze superiori ai 15 Hz: per ottenere questo risultato si può utilizzare un polo con una frequenza di spezzamento $\omega = 3$ Hz. Infatti ricordando che l'attenuazione ottenuta da questo tipo di filtro è di 20 dB per decade, si avrà che il modulo dei segnali con frequenza maggiore di 15 Hz, verrà ridotto di almeno 10 volte rispetto al valore originale. La sua funzione di trasferimento sarà:

$$PB(s) = \frac{1}{1 + 0.333s} \tag{14}$$

mentre il diagramma di Bode è rappresentato in Figura 15.

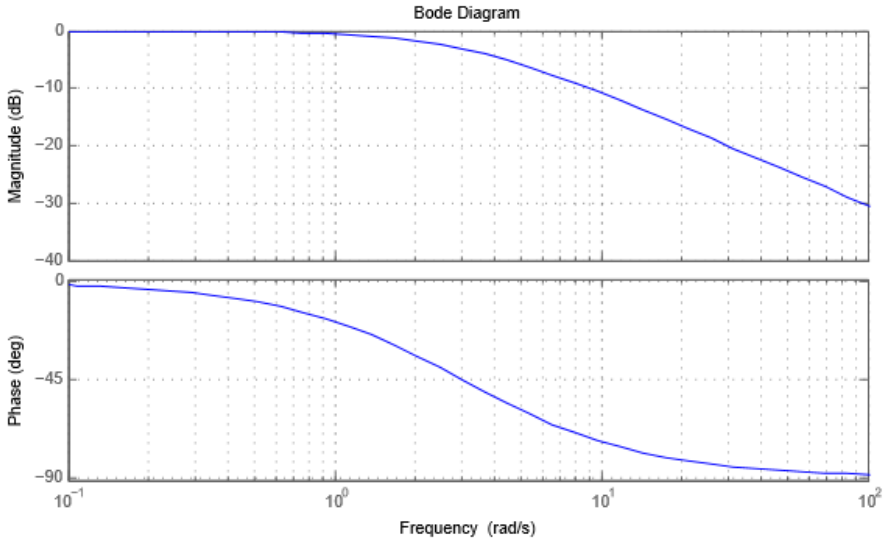

Figura 15: Diagramma di bode del filtro passa bassa

In Figura 16 invece possiamo vedere i miglioramenti per quanto riguarda il contenuto rumoroso del segnale, in rosso infatti è rappresentato il segnale filtrato mentre in blu il segnale originale.

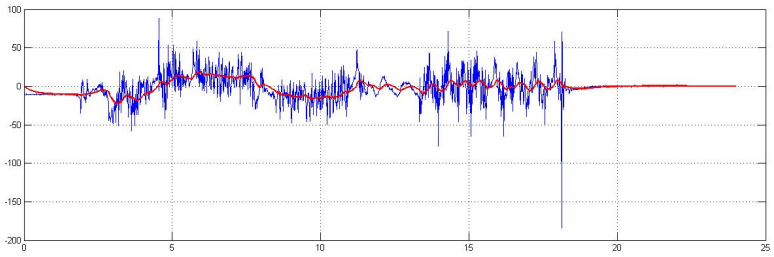

Figura 16: Il segnale di forza filtrato

In questo caso anche la velocità calcolata attraverso la simulazione in Matlab (Figura 17) sembra coerente con la forza in ingresso: possiamo infatti distinguere tre picchi di velocità ognuno corrispondente alle pressioni esercitate e descritte nel paragrafo 3.2.3. Durante l'esperimento però il manipolatore non si è comportato come ci si aspettava: la risposta del sistema alle sollecitazioni esterne non era accettabile: i movimenti avvenivano con molto ritardo e a volte non erano coerenti con la forza applicata. Inizialmente la causa di questo comportamento è stata imputata al filtro passa basso: esso agisce in modo da migliorare il contenuto del segnale di forza, ma allo stesso tempo introduce all'interno del sistema dei ritardi.
Perciò sono state valutate altre soluzioni per risolvere il problema: tra le varie alternative si è deciso di implementare un nuovo filtro con una frequenza di spezzamento più alta ($\omega = 15\,\text{Hz}$), per cercare

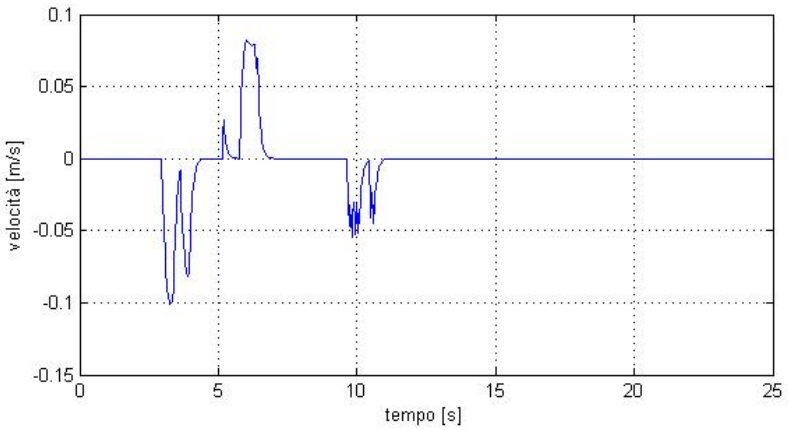

Figura 17: Velocità in uscita dal del modello con un filtro passa basso

di migliorare il tempo di risposta totale del sistema. All'interno del modello è stata quindi inserita la sua funzione di trasferimento:

$$PB(s) = \frac{1}{1 + 0.0666s}, \tag{15}$$

ma come si può notare dalla Figura 18, con tale frequenza l'azione del filtro sul segnale di forza non era sufficiente, e non ha portato ad un significativo miglioramento del comportamento totale del robot.

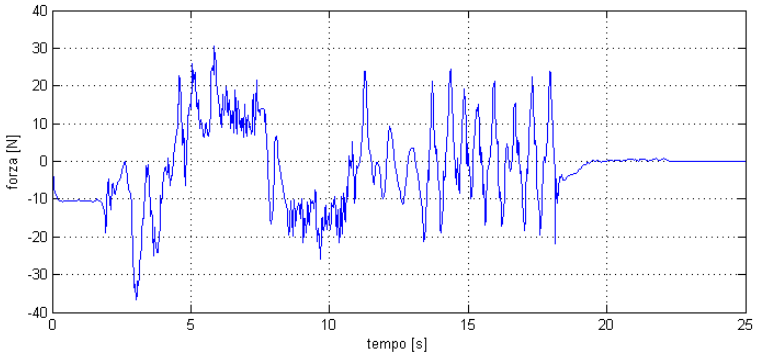

Figura 18: Segnale di forza dopo il filtraggio delle frequenze minori di 15 Hz

3.2.6 *Modello con filtro passa basso all'interno dell'anello di retroazione*

Un'altra prova è stata invece effettuate spostando il filtro passa basso all'interno dell'anello di retroazione, in modo da lavorare direttamente sui valori di accelerazione e di conseguenza sulla velocità del sistema. Come conseguenza è stata modifica la funzione di trasferimento dell'intero sistema:

$$W(s) = \frac{1}{K_d} \frac{1}{1 + \frac{s}{K_d} + \frac{\tau}{K_d} s}, \tag{16}$$

dove ricordiamo che τ è la costante di tempo del filtro, mentre K_d il guadagno di velocità. La (16) rappresenta un sistema dinamico del secondo ordine [3], sempre stabile in quanto i suoi poli sono delle radici immaginarie a parte reale positiva: vengono quindi introdotte nella risposta a regime delle oscillazioni, che devono essere contenute agendo sui valori di τ e K_d. Come si può notare dalla Figura 19 il risultato simulato con Matlab non è stato neanche in questo caso soddisfacente, e anche l'esperimento con questa configurazione non ha avuto gli effetti sperati.

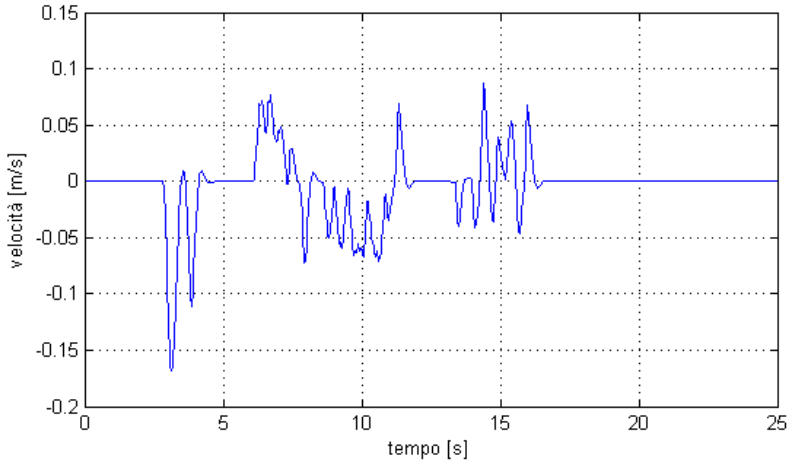

Figura 19: Velocità in uscita dal modello dinamico del secondo ordine

3.2.7 *Modello con filtro passa basso e zero-order-hold*

Come conseguenza delle prove precedentemente eseguite, si è arrivati alla conclusione che in alcuni casi questi risultati non corretti erano dovuti ad fattori esterni al modello realizzato; questi impedivano al robot di lavorare correttamente ricevendo in ingresso dei comandi con una frequenza elevata. Infatti negli esperimenti svolti fin qui i valori di velocità e accelerazione, che venivano ricavati attraverso la stima ottenuta con il modello, sono sempre stati elaborati ed inviati con una frequenza pari a quella di acquisizione, che ricordiamo era di 125 Hz. Nell'ultima prova invece è stato deciso di campionare il segnale di forza, filtrato utilizzando (14), con un tempo di campionamento $Tc = 0.12\,s$, 15 volte maggiore rispetto al tempo di usato in precedenza. Per fare questo è stato utilizzato in Simulink il blocco zero-order-hold, il quale campiona il segnale con l'intervallo temporale specificato, e mantiene il valore trovato per tutta la durata di tale intervallo. L'uscita di velocità calcolata attraverso Matlab e Simulink e rappresentata in Figura 20; è risultata infine coerente con il segnale di forza applicato. Tuttavia in questo caso anche il comportamento del robot registrato durante la prova è risultato adeguato: il manipo-

latore rispondeva in modo corretto alle forze esercitate dall'esterno, si fermava quando terminava l'interazione, anche se era comunque presente un ritardo nell'ordine di 0.2 s nella risposta complessiva del sistema.

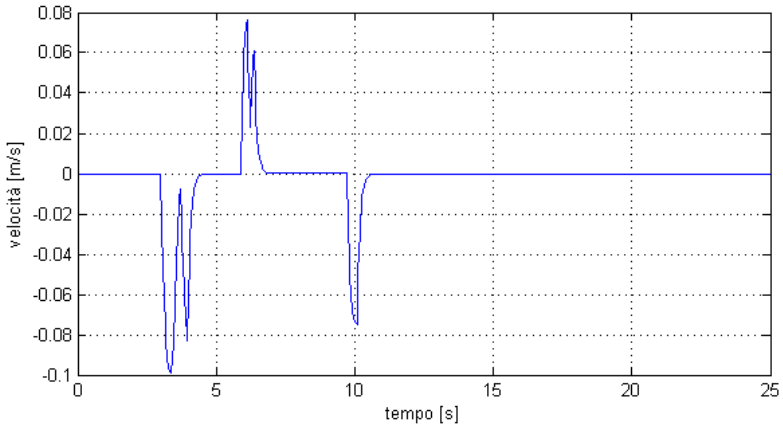

Figura 20: Velocità in uscita dal modello con un filtro passa bassoo e zero-order-hold

3.2.8 *Modello in Simulink del robot manipolatore*

Il modello completo realizzato in Simulink e utilizzato per la simulazione dell'esperimento è rappresentato in Figura 21. Per il suo corretto funzionamento è stato inoltre utilizzato il seguente script Matlab:

```
Fe=xlsread('Forze.xlsx');     %Array di forze caricato da
                              excel

Fs=125                        % Frequenza di campionamento
T=1/Fs                        % periodo di campionamento
L=3000                        % Lunghezza del segnale
                              (3000 campioni)
t=(0:L-1)*T;

F=timeseries(Fe,t);

Kd=200;                       %guadagno di velocità
M=20;                         %massa in Kg
```

Si ha quindi che i campioni di forze reali prelevati dalla prova e salvati in un file Excel vengono caricati nel vettore Fe. Viene inoltre creato il vettore temporale t con intervalli temporali di 0.008 s, che insieme ad Fe, grazie alla funzione *timeseries*, andrà a formare la serie temporale F. Questa viene introdotta nel modello Simulink attraverso il blocco

from workspace: i valori di forza vengono prima filtrati e campionati, poi prima di entrare all'interno del sistema in retroazione, avviene l'eliminazione della componente continua con il metodo spiegato precedentemente (Paragrafo 3.2.5). La forza viene in seguito confrontata con il riferimento proveniente dalla retroazione, il conseguente errore viene diviso per la massa M, inizializzata nello script, per calcolare l'accelerazione. Si ha quindi il saturatore di accelerazione (Paragrafo 3.2.1) e l'integrazione per il calcolo della velocità. Quest'ultima viene moltiplicata per il guadagno K_d, anche esso definito e inizializzato nello script, per chiudere l'anello di retroazione. Infine troviamo il saturatore di velocità che invia i dati finali dell'uscita all'ultimo blocco *to workspace*, affinché vengano salvati nel vettore *v*.

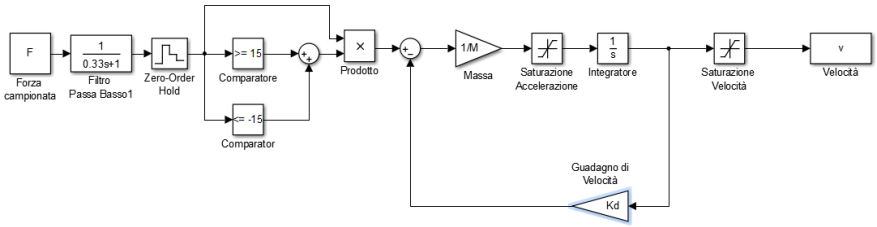

Figura 21: Rappresentazione in Simulink del modello completo

4

IMPLEMENTAZIONE DEL SISTEMA

In questo capitolo viene spiegato il modo in cui sono stati implementati, attraverso il linguaggio di programmazione C++, i diversi blocchi che formano il modello del sistema. Inoltre inizialmente viene descritto brevemente il funzionamento di ROS, per illustrare dove il codice sviluppato si inserisce all'interno del sistema.

4.1 INTRODUZIONE A ROS

ROS è un framework, cioè un'architettura logica di supporto che fornisce diverse funzionalità, le quali possono essere modificate dagli utenti in modo da ottenere un software specifico per una determinata applicazione. Inoltre i framework possono includere al loro interno programmi di sostegno, compilatori, librerie di codice, set di strumenti e API (*Application programming interfaces*), che insieme consento lo sviluppo di un progetto o di un sistema. L'unità di organizzazione del software in ROS sono i *package*, i quali possono contenere file eseguibili, scripts ma anche librerie. Un particolare tipo di eseguibile è il *nodo* la cui caratteristica fondamentale è la capacità di poter comunicare con altri nodi. Questi infatti attraverso determinate architetture di comunicazione (*publisher e subscriber*), possono pubblicare dei messaggi relativi ad un determinato *topic*, oppure iscriversi a un topic per ricevere un messaggio, condividendo di conseguenza delle informazioni. Si è deciso quindi di implementare il codice per sviluppare il modello proposto all'interno di un nodo con la capacità di leggere i valori di forza pubblicati in un determinato topic. Questa azione e l'esecuzione dell'algoritmo sviluppato avvengono in modo ciclico: durante ogni iterazione, letta la forza applicata, viene calcolata la velocità che deve avere l'end effector e vengono salvati i dati utili a effettuare la retroazione del segnale nel passo successivo.

4.2 VARIABILI E COSTANTI UTILIZZATE

Di seguito verranno analizzate le vari porzioni di codice C++ utilizzate per implementare il modello. Per capirle in modo corretto è innanzitutto necessario introdurre le diverse variabili inizializzate e le loro relative spiegazioni :

```
const int M = 20;            /* massa*/
const double Kd = 200;       /* guadagno di velocità */
const double Tc = 0.008;     /* tempo di campionamento in
                             secondi */
```

```
double f_i = 0.0;            /* forza filtrata lungo l'asse
                             z istante attuale */
double f_0 = 0.0;            /* forza in retroazione lungo
                             l'asse z istante precedente */
double f_e = 0.0;            /* errore di forza */
double f_w = 0.0;            /* forza ricavata sottoscriven-
                             dosi al topic wrench */
double f_i_old = 0.0         /* forza filtrata all'istante
                             precedente
double f_w_old = 0.0;        /* forza all'istante precedente
                             ricavata sottoscrivendosi al
                             topic wrench*/
double a_i = 0.0;            /* accelerazione attuale */
double a_0 = 0.0;            /* accelerazione istante
                             precedente */

double v_i = 0.0;            /* velocità  attuale */
double v_0 = 0.0;            /* velocità  istante preceden-
                             te */
```

4.3 IL MODELLO

Per l'implementazione del modello vero e proprio il codice utilizzato è il seguente:

```
f_0=Kd*v_0;          /* calcolo forza precedente */
f_e=f_i-f_0;         /* calcolo errore di forza */
a_i=f_e/M;           /* calcolo accelerazione attuale */
```

Inizialmente viene calcolato il valore della forza proveniente dall'anello in retroazione, moltiplicando per il guadagno di velocità la velocità calcolata all'istante precedente. In seguito tale valore è utilizzato per calcolare l'errore di forza, il quale a sua volta servirà per ottenere l'accelerazione. A questo punto viene implementato l'integrale per il calcolo della velocità attuale attraverso la regola del trapezio: considerando l'intervallo di tempo tra un campione k e il precedente $k-1$, viene innanzitutto calcolata l'area formata dal trapezio individuato dai valori di $a(k-1)$ e $a(k)$, visibile in Figura 22, attraverso il seguente codice:

```
double AreaIntegrale (double y1, double y2)
  {
    return (y1+y2)*0.5*Tc;
  }
```

In seguito a quest'area viene aggiunta la somma delle aree calcolate durante gli istanti precedenti, per ottenere il valore istantaneo di velocità:

```
v_1=v_0+AreaIntegrale(a_0,a_i);
```

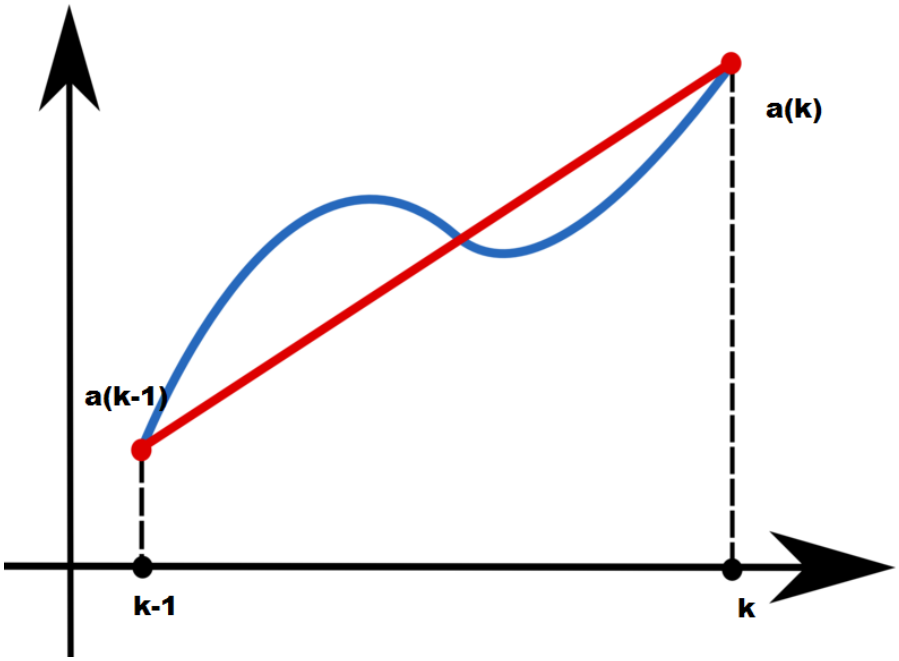

Figura 22: Metodo di integrazione trapezoidale

Questo viene assegnato all'end effector attraverso una funzione già implementata all'interno della libreria

```
\emph{rur53_manipulation/arm_move.h},
```

sviluppata dagli utenti del laboratorio, permettendo così di sostituire l'architettura di comunicazione che forma il publisher con il seguente codice:

```
#include <rur53_manipulation/arm_move.h>
#include <vector>

std::vector<double> vel(6, 0.0);
    vel[2] = v_i;

arm->execSpeedControl(vel,"/ur5_arm_base", a_i, Tc);
```

La funzione prevede che in ingresso le sia fornito:

- un vettore contenente le velocità cartesiane e angolari. In questo caso visto che stiamo trattando l'asse z andremo soltanto a modificare il terzo valore del vettore (vel[2]).
- Il frame rispetto al quale vengono fornite le informazioni di velocità e forza.
- L'accelerazione che deve avere l'end effector.
- L'intervallo temporale che indica per quanti secondi deve essere mantenuta tale velocità.

4.4 I SATURATORI

Per l'implementazione dei saturatori sono stati utilizzati delle semplici istruzioni condizionali con i dovuti valori:

```
if(a_i > 1){
      a_i = 1;
    }
else if(a_i < -1){
      a_i = -1;
    }

if(v_i > 0.5){
      v_i = 0.5;
    }
else if(v_i < -0.5){
      v_i = -0.5;
    }
```

4.5 IL FILTRO PASSA BASSO

Per la realizzazione del filtro passa basso in ambito digitale innanzitutto è stato necessario fare una discretizzazione della sua funzione di trasferimento $PB(s)$ (Eq. 14) attraverso una funzione Matlab, con tempo di campionamento pari a 0.12 s:

```
PBD=c2d(PB,0.12);
```

Il risultato di tale operazione è una trasformata Zeta che lega i campioni in ingresso e uscita al filtro:

$$\frac{Y(z)}{U(z)} = \frac{0.02374z^{-1}}{1 - 0.9763z^{-1}}; \tag{17}$$

Da quest'ultima equazione moltiplicando per $\frac{U(z)}{1-0.9763z^{-1}}$ sia a destra che a sinistra, e ricordando che la moltiplicazione per z^{-1} corrisponde all'introduzione di un ritardo temporale pari al tempo di campionamento, troviamo la seguente relazione:

$$y(k) = 0.9763y(k-1) + 0.02374u(k-1), \tag{18}$$

dove:

- $y(k)$ corrisponde al valore di forza filtrato all'istante attuale.
- $y(k-1)$ corrisponde al valore di forza filtrato all'istante precedente.
- $u(k-1)$ corrisponde al valore di forza non filtrato all'istante precedente.

Definite queste uguaglianze e ricordando i nomi delle diverse variabili, possiamo infine introdurre il codice necessario per l'implementazione del filtro:

```
f_i=f_i_old*0.9763+f_w_old*0.02374;
```

4.6 I COMPARATORI

I comparatori utilizzati per eliminare la componente continua sono stati implementati analogamente ai saturatori, con una semplice istruzione condizionale:

```
if(f_i > -15 && f_i < 15)
    {
          f_i = 0.0;
    }
```

Questa porzione di codice deve esser inserita dopo il calcolo del valore di forza filtrato, permettendo così di annullare correttamente i valori di forza che non rispettano il limite imposto

4.7 L'ANELLO DI RETROAZIONE

Per permettere il funzionamento dell'anello di retroazione e quindi una corretta iterazione dell'istante successivo, alla fine del codice proposto vengono aggiornati i valori della velocità, accelerazione e forza degli istanti precedenti, attraverso il codice seguente:

```
v_0=v_i;           /* imposto il nuovo valore di v_0 */
a_0=a_i;           /* imposto il nuovo valore di a_0 */
f_w_old=f_w;       /* imposto il nuovo valore di f_w */
f_i_olf=f_i;       /* imposto il nuovo valore di f_i */
```

Infine bisogna specificare che oltre a questi elementi, il codice contiene anche l'architettura utilizzata per iscriversi al topic in cui viene pubblicato il messaggio, contenete la forza stimata al TCP del robot.

5

CONCLUSIONI

La realizzazione del progetto si è rivelato un lavoro ricco di compromessi e non privo di imprevisti. La modelizzazione del robot è stata fatta in maniera semplificata, questo però ha creato delle approssimazioni e quindi delle differenze tra le simulazioni e la realtà. Il risultato finale ottenuto può comunque ritenersi accettabile: il robot rispondeva in modo corretto alle sollecitazioni esterne, la risposta del sistema era accettabile per il task programmato. Tuttavia erano comunque presenti dei ritardi che, dopo uno studio accurato, potrebbero comunque esser eliminati. Inoltre l'esperimento è stato utile per prendere confidenza con le dinamiche introdotte dall'anello di retroazione della velocità presente nei controlli di forza indiretti, e con i problemi che si riscontrano in ogni prova pratica. I sviluppi futuri dell'esperimento potrebbero comprendere un'analisi più approfondita del modello, con l'introduzione di un parametro che potesse rappresentare la rigidezza del robot. Si sarebbe potuto così studiare il sistema nella sua completezza introducendo un anello di retroazione della forza, per effettuare un controllo più preciso e accurato.

BIBLIOGRAFIA

[1] Bernier Catherine. What's the difference between robotiq activedrive and ur freedrive? http://dof.robotiq.com/discussion/78/whats-the-difference-between-robotiq-activedrive-and-ur-freedrive, 2016.

[2] Edwin A Erlbacher. Force control basics. *Industrial Robot: An International Journal*, 27(1):20–29, 2000.

[3] Giovanni Marro. *Controlli automatici*. Zanichelli, 1978.

[4] Matthew T Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(6):418–432, 1981.

[5] Kiyoshi Ohishi, Masaru Miyazaki, and Masahiro Fujita. Hybrid control of force and position without force sensor. In *Industrial Electronics, Control, Instrumentation, and Automation, 1992. Power Electronics and Motion Control., Proceedings of the 1992 International Conference on*, pages 670–675. IEEE, 1992.

[6] SI Part. Impedance control: An approach to manipulation. *Journal of dynamic systems, measurement, and control*, 107:17, 1985.

[7] J Kenneth Salisbury. Active stiffness control of a manipulator in cartesian coordinates. In *Decision and Control including the Symposium on Adaptive Processes, 1980 19th IEEE Conference on*, volume 19, pages 95–100. IEEE, 1980.

[8] Bruno Siciliano and Luigi Villani. *Robot force control*, volume 540. Springer Science & Business Media, 2012.

[9] Berryman Tyler. What is the new robotiq activedrive mode on universal robots? http://dof.robotiq.com/discussion/79/what-is-the-new-robotiq-activedrive-mode-on-universal-robots, 2016.

[10] Daniel E Whitney. Force feedback control of manipulator fine motions. *Journal of Dynamic Systems, Measurement, and Control*, 99(2):91–97, 1977.

[11] Daniel E Whitney. Historical perspective and state of the art in robot force control. *The International Journal of Robotics Research*, 6(1):3–14, 1987.