Università degli Studi di Padova

Dipartimento di Tecnica e Gestione dei sistemi industriali

Corso di Laurea Magistrale in Ingegneria Meccatronica

Funzionalità e strategie di controllo dei robot collaborativi

Relatore
Chiar.mo Prof. Giovanni Boschetti

Candidato
Omar Antoniazzi
2004170

Anno Accademico 2022–2023

Sommario

Nel corso degli anni, grazie all'impiego di nuove tecnologie, i robot collaborativi che consentono agli umani di lavorare assieme a loro, sono sempre più presenti nel mondo del lavoro, portando sicuramente molti benefici per aziende e lavoratori. Questo nuovo aspetto però comporta che l'operatore umano sia formato in maniera specifica su questi robot. Questo elaborato ha l'obiettivo di fornire una panoramica delle varie funzioni collaborative che possono essere eseguite dai cobot e testare alcuni tipi di controllo sul Fanuc CR-15iA, braccio antropomorfo a 6 assi, che gli consentiranno di essere manovrato tramite il polso del robot con una funzione chiamata Manual Guided Teaching. Questa può essere sviluppata con diversi tipi di controllo ed è questo l'ambito in cui si svilupperà questa tesi. Grazie a questa applicazione sarà possibile anche per alcuni lavoratori non particolarmente avvezzi alla programmazione robotica di riuscire ad insegnare al braccio robotico un percorso da seguire semplicemente prendendolo per il polso e trascinandolo nei punti che vogliono siano attraversati dal robot in maniera molto intuitiva, limitando l'uso del Teach Pendant alla sola attivazione del programma. Si analizzerà sia il cobot usato per i test sia i metodi e le strategie di controllo che permetteranno una guida manuale di questo. Tutto questo per capire fino a che punto possa arrivare questo robot nella collaborazione con un operatore umano in ambito industriale.

Ringraziamenti

Vorrei riservare questo spazio della mia tesi di laurea ai ringraziamenti verso tutti coloro che hanno contribuito e mi hanno accompagnato durante questi anni di studi. Per prima cosa, vorrei ringraziare il mio relatore Giovanni Boschetti, che mi ha concesso l'opportunità di lavorare a questo progetto. Un grazie speciale alla mia famiglia, grazie ai loro sacrifici, grazie al loro importante supporto durante tutto il mio percorso universitario, grazie per aver creduto in me. Grazie a tutti i miei amici, che vicini o lontani mi hanno sempre incoraggiato, condividendo con me molti momenti, alleggerendomi la vita con la loro compagnia. Un grazie anche a tutte le persone che hanno contribuito a formarmi come persona durante tutti questi anni ed arrivare a questo traguardo che segnerà sicuramente un punto di svolta nella mia vita. Grazie.

Indice

1 Introduzione **11**
- 1.1 Cobot . . . 11
  - 1.1.1 Tipi di collaborazione . . . 12
- 1.2 Fanuc . . . 13
- 1.3 Fanuc CR-15iA . . . 14
  - 1.3.1 Elementi Principali Cobot Fanuc . . . 16
  - 1.3.2 Sensore di forza . . . 17
  - 1.3.3 Pinza schunk co-act EGP-C . . . 19
- 1.4 Modalità di funzionamento . . . 20
  - 1.4.1 T1(<250mm/sec) e T2(100%) . . . 21
  - 1.4.2 Auto mode . . . 21

2 Funzioni Collaborative Base **23**
- 2.1 Set Up . . . 23
- 2.2 Funzioni collaborative . . . 26
  - 2.2.1 Contact stop function . . . 26
  - 2.2.2 Retreat after contact stop . . . 27
  - 2.2.3 Push to escape . . . 27
  - 2.2.4 Manual Guided Teaching . . . 28

3 Strategie di Controllo **31**
- 3.1 Stabilità . . . 32
- 3.2 Controllo Proporzionale . . . 33
- 3.3 Controllo di ammettenza con modello di oscillatore semplice smorzato . . . 34
  - 3.3.1 Modello oscillatore semplice smorzato . . . 36
- 3.4 Controllo Mixed Proporzionale e di Ammettenza . . . 40

4 Programmi Implementati di Manual Guided Teaching **43**
- 4.1 Basi del linguaggio Karel di Fanuc . . . 43
- 4.2 Controllo Proporzionale . . . 45
- 4.3 Controllo di ammettenza modello oscillatore semplice smorzato . . . 48
- 4.4 Controllo Mixed Proporzionale e di Ammettenza . . . 52

5 Risultati sperimentali **53**
5.1 Risultati Controllo Proporzionale 54
5.2 Risultati Controllo di Ammettenza 54
5.3 Risultati Controllo Mixed 56

6 Conclusioni **57**

Elenco delle figure

1.1 Tipi di collaborazione umano-robot 13
1.2 Fanuc CR-15iA 15
1.3 Spazio di lavoro del Fanuc CR-15iA 15
1.4 Controllore e TP del robot collaborativo Fanuc 16
1.5 Sensore di forza FS-15iA 18
1.6 Codice per acquisire i valori letti dal sensore di forza nei registri . 19
1.7 Pinza schunk co-act EGP-C 19
1.8 Three mode switch 21

2.1 Menù dei limiti delle forze esterne 24
2.2 Menù dei set up predefiniti del payload 25
2.3 Movimento dei giunti J1 e J2 nel push to escape 28

3.1 Schema a blocchi della stabilità accoppiata tra umano e robot . . 33
3.2 Schema a blocchi semplificato di un controllo di ammettenza disaccoppiato 35
3.3 Dinamica oscillatore smorzato ad anello aperto 37
3.4 Schema oscillatore semplice smorzato 38

5.1 Forza usata come riferimento 53
5.2 Velocità del controllo proporzionale 54
5.3 Velocità del controllo di ammettenza con rigidezza k=0 55
5.4 Velocità del controllo di ammettenza con rigidezza k=20 55
5.5 Velocità del controllo mixed 56

Capitolo 1

Introduzione

1.1 Cobot

Il mercato dei robot industriali negli ultimi anni ha cominciato a richiedere, oltre alla personalizzazione di massa e tempi di consegna breve, sistemi di assemblaggio flessibili, poco costosi e polivalenti. Proprio per questo, soprattutto nelle piccole e medie imprese i robot tradizionali cominciano ad andare in disuso. Una delle alternative più valide sono i robot collaborativi, detti anche cobot, pensati per interagire in maniera sicura con un operatore umano in uno spazio condiviso [**1**]. I più comuni sono quelli che fanno uso di EOAT (End-of-arm tooling), dei robot che utilizzano la tecnologia di fine braccio, ovvero quelli che hanno attaccato al polso lo strumento finale che gli consente di svolgere il compito che gli è stato assegnato. Ci sono vari aspetti per i quali i cobot si differenziano dai classici robot industriali. Innanzitutto sono facilmente trasportabili e la maggior parte può essere montato su superfici sia orizzontali che verticali. Al contrario, i robot classici sono più potenti e ingombranti, rendendo difficile anche progettare il layout del processo produttivo. Inoltre seguono un programma fisso che svolgono senza badare agli ambienti circostanti. Sono quindi necessarie delle barriere che impediscano ai robot di danneggiare, in caso di malfunzionamento o calibrazione fatta male, persone e oggetti intorno a loro. I cobot invece non necessitano di questi recinti e consentono alle persone di lavorare con loro senza alcun rischio. Per esempio si può scegliere se al minimo contatto con l'operatore, questi si immobilizzino o si ritirino, tutto grazie a sensori e attuatori che lo fanno lavorare ad una velocità consona e controllano in maniera appropriata la loro forza.

Inoltre, sono molto affidabili e imparano velocemente le tasks che hanno la possibilità teorica di compiere, possono quindi essere usati anche in ambienti pericolosi, dove l'operatore umano potrebbe farsi del male, infatti con l'aumento dei cobot, si è notata una diminuzione del numero di incidenti sul luogo di lavoro.
Un altro vantaggio è la facilità di programmazione che hanno, infatti a differenza dei robot tradizionali che richiedono spesso competenze specifiche, questi possono riuscire a riprodurre i movimenti mostratogli dall'operatore, superando quindi la difficoltà della programmazione che alcuni operatori non formati possono presentare. Proprio grazie a questa facilità di utilizzo riescono a risolvere molte difficoltà legate all'assemblaggio, infatti interagendo con l'umano riducono il carico di lavoro per quest'ultimo durante tutto l'arco della giornata.
Infine i sistemi collaborativi permettono di diminuire i costi unitari di produzione, infatti in base al tipo di mansione che si vuole far svolgere tra assembly, placement, handling e picking si osserva che un alto indice di collaborazione umano-robot ha un grande impatto sulla produttività nella maggior parte dei casi. In particolare i robot collaborativi sono molto vantaggiosi negli incarichi di assembly, si comportano discretamente per picking e placement anche se i robot tradizionali sono più precisi e veloci in quegli ambiti.
Da questo si evince che l'evoluzione verso un sistema collaborativo tra umano e robot è dettata prevalentemente da ragioni economiche, efficienza dello spazio e salute sul posto di lavoro. Negli ultimi anni i cobot sono specializzati in compiti specifici che imparano direttamente nell'ambiente di lavoro, a volte anche in autonomia. Inoltre, in alcuni casi i robot collaborativi potranno comunicare tra di loro, aumentando efficienza e produttività.

1.1.1 Tipi di collaborazione

La collaborazione umano-robot è un argomento molto ampio e quindi facile da fraintendere. Un articolo della rivista Robotics[**2**] riguardante questa tematica propone una classificazione delle differenti metodologie nelle quali umani e robot potessero interagire tra loro.

In figura 1.1 sono rappresentati i vari stadi di interazione, da quello tradizionale a sinistra, nel quale il robot è racchiuso in una cella e quindi assenza di collaborazione, a quello finale a destra che illustra un ambiente collaborativo.

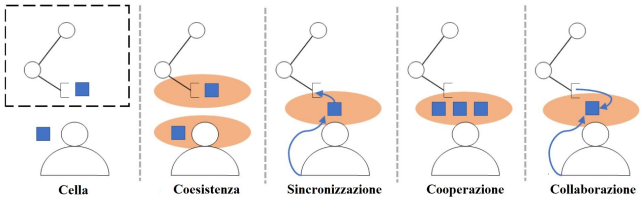

Figura 1.1: Tipi di collaborazione umano-robot

- Coesistenza: Quando operatore umano e robot sono nello stesso ambiente ma generalmente non interagiscono tra di loro.
- Sincronizzazione: Quando operatore umano e robot lavorano nello stesso spazio ma in intervalli di tempo diversi.
- Cooperazione: Quando operatore umano e robot lavorano nello stesso spazio allo stesso tempo ma hanno incarichi diversi.
- Collaborazione: Quando operatore umano e robot eseguono lo stesso incarico insieme, quindi l'azione dell'uno influenza le azioni dell'altro.

Si tiene a precisare che questa è solo una delle classificazioni possibili, ma è stata scelta per la chiarezza con cui suddivide i vari stadi della collaborazione.
In questo elaborato verranno trattate le funzioni collaborative di base della maggior parte dei cobot, inoltre sarà spiegato come implementare diversi tipi di Manual Guided Handed nel caso ne fosse sprovvisto. L'implementazione in questo caso è stata fatta per un Fanuc CR-15iA.

1.2 Fanuc

FANUC è l'acronimo di Factory Automation NUmerical Control ed è un gruppo societario capeggiato da FANUC corporation, Fanuc America corporation e Fanuc Europe Corporation S.A. Si occupa della produzione di robot e controlli numerici computerizzati volti prevalentemente ad ottimizzare i processi di produzione nelle aziende.
Fondata nel 1956 dal Dr.Seiuemon Inaba, uno dei primi ad occuparsi di concetti come il controllo numerico, dopo più di sessant'anni di attività Fanuc è uno dei principali produttori a livello mondiale

nel suo ambito. Ad oggi le industrie di meccanica, specialmente nel settore dell'automazione, che usano tecnologie fanuc sono molte e sempre in aumento, questo perché ha un'ampia gamma di prodotti per l'automazione industriale che riescono a soddisfare le esigenze del cliente, fornendo inoltre un buon bilanciamento tra il costo totale di proprietà e i risultati di produzione.
Nella vasta gamma di prodotti sviluppati da Fanuc si trovano:

- Robot:Variano si a livello di grandezza che di carico che possono trasportare, attualmente si contano più di 100 modelli, riuscendo a coprire molte applicazioni e settori differenti grazie anche alle opzioni di personalizzazione per adattarsi ad ogni situazione. Per la stesura di questo elaborato si è fatto uso del Fanuc CR-15iA, uno dei loro robot collaborativi.

- Sistemi CNC: Spazia da controlli a servomotori a mototori a pacchetti di semplice installazione.

- Sistemi Laser

- Macchine per stampaggio a inezione

Tutti questi prodotti sono sviluppati, realizzati e collaudati internamente alla società.

1.3 Fanuc CR-15iA

Il robot che si andrà ad utilizzare per i vari test è il fanuc CR-15iA in figura 1.2, un robot collaborativo antropomorfo a 6 assi.
Questo cobot a differenza di altri che utilizzano dei sensori esterni, gestisce tutto grazie ad un singolo sensore di forza incorporato nella sua base. Questo lo rende molto robusto e affidabile, anche se ovviamente deve essere fissato a terra o su qualche superficie che gli consenta lo stesso grado di robustezza. La possibilità di essere montato in posizione verticale, rovesciato o sul muro orizzontalmente lo rende molto flessibile e versatile, inoltre il piedistallo è abbastanza compatto in modo da non occupare troppo spazio. Nonostante ciò il fanuc può comunque raggiungere 2413mm in corsa verticale e 1441mm in quella orizzontale, come si può vedere in figura 1.3 dov'è rappresentato lo spazio di lavoro. Invece per quanto riguarda il carico sostenibile dal polso, detto payload, questo modello può arrivare fino a 15kg. Infine è doverosa una menzione al software DCS (Dual

Figura 1.2: Fanuc CR-15iA

Check Safety) che verrà trattato in seguito, una tecnologia utilizzata sulla maggior parte cobot fanuc, serve a monitorare la posizione e la velocità del robot per assicurare la sicurezza dell'operatore umano.

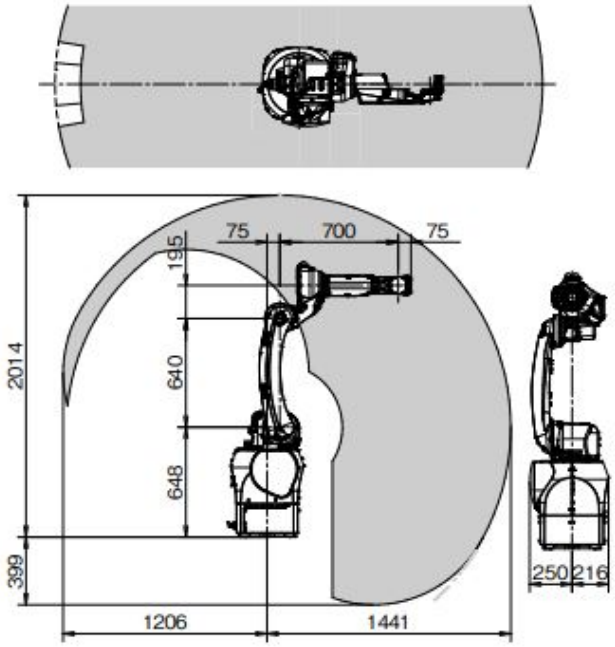

Figura 1.3: Spazio di lavoro del Fanuc CR-15iA

1.3.1 Elementi Principali Cobot Fanuc

Un tipico cobot Fanuc comprende sempre tre principali elementi, un braccio robotico, un controllere e un teach Pendant(Figura 1.4). Questi sono standard e sono essenziali per qualsiasi tipo di braccio antropomorfo cobot Fanuc. Possono essere personalizzati in base alle esigenze del cliente, in particolare il braccio robotico, sempre nei limiti della robotica collaborativa e in base al budget che si possiede. Di seguito sarà fornita una descrizione di questi tre elementi fondamentali.

Figura 1.4: Controllore e TP del robot collaborativo Fanuc

Braccio Robotico

Un macchinario articolato con servo motori che si occupa di eseguire il lavoro fisico programmato. Venduto solitamente senza tool di fine braccio (EOAT) può essere integrato con essi in base alle necessità che comporta la mansione da eseguire. Tramite cablaggio è possibile far passare informazioni dal tool, al braccio robotico e infine al controllore, che li analizza e li utilizza come vuole il programma, in questo elaborato per esempio, si utilizzerà un sensore di forza che passerà al controllore informazioni riguardo forze e momenti che agiscono nello specifico sul polso del robot. In alcuni di questi modelli si fa anche uso di aria pressurizzata per attivare alcuni output come le pinze o ventose e far prendere oggetti al braccio.

Controllore

Questo consente di controllare il braccio robotico, l'alimentazione, gli strumenti ausiliari e le varie personalizzazioni che possono essere richieste. Inoltre data la grande varietà di applicazioni riguardanti l'automazione, i controllori devono contenere gli hardware necessari che consentano al robot di essere utilizzato in diverse applicazioni o assieme ad altre tecnologie. Un'altra importante funzione del controllore è occuparsi dei vari limiti di posizione che il braccio robotico non può oltrepassare o eventuali pose del robot non ammesse perché porterebbero a singolarità, nel caso in cui il controllore percepisse che continuando a muoversi in una direzione venisse violato uno di questi limiti, farebbe arrestare immediatamente il braccio robotico.

Teach Pendant

Questa è la principale interfaccia tra robot e operatore durante l'insegnamento di programmi o la manutenzione. Durante la modalità automatica che verrà trattata di seguito invece, il TP viene solitamente lasciato in disparte in quanto il robot sarà in grado di eseguire il programma in maniera autonoma. Il TP solitamente ha sia un touch screen che una tastiera fisica per visualizzare ed eventualmente editare i programmi creati da PC o direttamente dal TP. Ha inoltre un pulsante deadman switch che deve essere tenuto premuto ogni volta che si intende utilizzare il robot durante la teaching mode, altra modalità che verrà approfondita più avanti, in caso contrario il robot si arresterà. Tutto questo per assicurarsi che durante l'insegnamento o i test di un nuovo programma l'addetto che potrebbe operare vicino al robot sia sempre al sicuro semplicemente rilasciando il pulsante.

1.3.2 Sensore di forza

Come detto in precedenza, il modello Fanuc CR-15iA ha un sensore di forza incorporato nella base, questo gli consente di avere il senso del tatto senza bisogno di hardware o software ausiliari. Le funzioni collaborative di base sono facilmente realizzabili utilizzando solo questo sensore semplicemente seguendo i passaggi necessari per attivarle, però se si vuole fare qualcosa di più complesso come il manual guided teaching oppure far trasportare un oggetto al robot assieme all'operatore umano, dosando quindi la sua forza in

base a quella dell'uomo, è necessario ricorrere ad un altro sensore di forza. In questo caso, si ha il Force Sensor FS-15iA, mostrato in figura 1.5. Si è scelto di montarlo tra la fine del polso e il tool del braccio, questa è una delle due principali configurazioni dei sensori di forza detta Hand Mount Sensor, l'altra ugualmente molto valida in altri casi invece è detta Fixed Mount Sensor dove si fissa il sensore direttamente alla postazione di lavoro. Questo componente consente di sviluppare funzioni che hanno bisogno di un controllo di forze e coppie molto più accurati rispetto a quelli ottenuti dal sensore incorporato nella base, in particolare, grazie ad esso si è in grado di rilevare con precisione forze e coppie applicate all'utensile terminale in 6 gradi di libertà. Rende quindi possibili operazioni di assemblaggio, contornatura, misurazione e molte altre, tutte con la possibilità di essere svolte assieme ad un operatore umano. Prima

Figura 1.5: Sensore di forza FS-15iA

di cominciare ad utilizzarlo però, bisogna installarlo ed effettuare la calibrazione. Per realizzare ciò, basterà seguire le istruzioni del manuale Fanuc riguardante il sensore di forza e creare un programma che faccia muovere il sensore con il tool attaccato in determinate posizioni, ovvero tenendo il braccio orizzontale spostare il sensore di forza da una posizione verticale con il tool che punta verso il basso, ad una orizzontale lungo il braccio e infine ad una verticale verso l'alto.
Una volta fatti tutti i procedimenti necessari, il sensore sarà pronto e si potranno leggere i suoi valori direttamente dalla teaching pendant oppure dai registri sui quali sono state salvate le forze Fx,Fy,Fz e i momenti Mx,My,Mz. Per leggere i valori acquisiti dal sensore durante l'esecuzione di un programma sono necessarie le righe di codice riportate in figura 1.6, che salveranno quei valori dal registro 33 al registro 38.

```
R[33:FX]=$CCC_GRP[1].$UT_FORCE[1]
R[34:FY]=$CCC_GRP[1].$UT_FORCE[2]
R[35:FZ]=$CCC_GRP[1].$UT_FORCE[3]
R[36:MX]=$CCC_GRP[1].$UT_FORCE[4]
R[37:MY]=$CCC_GRP[1].$UT_FORCE[5]
R[38:MZ]=$CCC_GRP[1].$UT_FORCE[6]
```

Figura 1.6: Codice per acquisire i valori letti dal sensore di forza nei registri

Bisogna tenere in considerazione che questo codice salverà nei registri solo per poco tempo le forze registrate, prima di aggiornarle con quelle sempre più attuali, bisognerà quindi creare un loop grazie al quale le forze verranno salvate e utilizzate durante l'esecuzione del programma in base al tipo di funzione che vogliamo fargli svolgere così da sfruttare al meglio il sensore.

1.3.3 Pinza schunk co-act EGP-C

Il Fanuc CR-15iA del laboratorio in cui sono stati fatti tutti i test, è dotato di un EOAT chiamato Pinza schunk co-act EGP-C (Figura 1.7).

Questa è una pinza collaborativa elettrica utilizzata per prendere

Figura 1.7: Pinza schunk co-act EGP-C

componenti di piccole dimensioni. Dispone di due griffe parallele e viene azionata a 24V e I/O digitali. Grazie ad un trimmer, è

possibile regolare manualmente la forza di presa su quattro livelli (100%, 75%, 50%, 25%) dove il massimo sono 140N. Questo valore si riferisce alla somma aritmetica della forza di serraggio applicata a ciascuna ganascia e bisognerà regolarlo in base alla deformabilità degli oggetti che verranno presi. Purtroppo, non è possibile modificare la forza di della presa tramite programma, sarà quindi necessario fermare il robot per effettuare la modifica.
Per garantire la massima sicurezza è dotata di un limite integrato di corrente e un rivestimento di protezione dagli urti che potrebbero accadere durante il funzionamento collaborativo. Inoltre per sapere sempre se la posizione della pinza è "aperta" o "chiusa", essa possiede come output due sensori di prossimità induttivi integrati che collegati agli input RI[2] e RI[3] del robot forniscono le informazioni desiderate (RI[2]=ON pinza aperta, RI[3]=ON pinza chiusa). Inoltre è disponibile per una vasta gamma di cobot di diversi produttori, oltre a fanuc vale la pena menzionae anche KUKA e Universal Robots. Le griffe montate sono quelle standard vendute assieme alla pinza ma è possibile fabbricarne altre in base al compito che si vuole far svolgere al robot grazie ad una stampante 3D. Una volta collegata, si potrà comandare tramite gli input della pinza stessa, che in questo caso sono DI_1 e DI_2, collegati agli output RO[7] e RO[8] del robot, i colori del led posizionato sulla pinza. Gli altri due input sono stati collegati a RO[5] e RO[6] e consentono di aprire e chiudere le griffe quando passano da 0 a 1.

1.4 Modalità di funzionamento

Questo cobot ha tre principali modi in cui può essere utilizzato. Una volta acceso dall'interruttore generale, sarà necessario scegliere in base alle proprie necessità il modo in cui vogliamo che il robot operi e selezionare la propria scelta tramite il Three Mode Switch mostrato in figura 1.8. Girando la manopola si potrà cambiare la modalità di funzionamento in qualsiasi momento ma se lo si fa durante l'esecuzione di un programma il robot si bloccherà e bisognerà fare un reset degli errori. Per utilizzarlo è necessario avere una chiave data in dotazione all'arrivo del robot che una volta inserita consentirà la rotazione dell'interruttore.

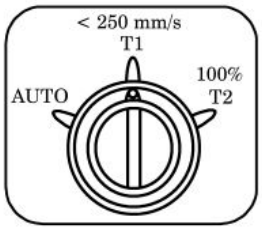

Figura 1.8: Three mode switch

1.4.1 T1(<250mm/sec) e T2(100%)

Prima modalità di prova del robot, detta test mode 1, serve principalmente per insegnare le varie posizioni in cui dovrà andare e controllare se esegue il percorso corretto a basse velocità. Tutti i programmi potranno essere eseguiti solo dal teach pendant e per tutta la durata dei test bisognerà tenere premuto il Deadman Switch, ovvero un interruttore a pressione sul retro del TP. Questo perché nel caso si verifichi qualche errore nel funzionamento del programma, il robot potrà essere arrestato semplicemente smettendo di applicare pressione al deadman switch, questa è solo una delle molteplici misure di sicurezza di questo cobot. La particolarità che la differenzia maggiormente dalla seconda modalità di test è che sia la velocità di jogging che quella durante l'esecuzione di un programma rimane sempre inferiore ai 250mm/sec.
La seconda modalità di funzionamento T2, ovvero Test mode 2, serve principalmente per effettuare le prove finali. Questo perché valgono tutte le considerazioni sulla sicurezza già spiegate in T1 con la differenza che in questo caso durante l'esecuzione dei programmi (quindi nel jogging si comporta nella stessa maniera) la velocità non è limitata. In questo modo si potrà verificare l'effettiva dinamica e percorso del braccio robot.

1.4.2 Auto mode

Questa modalità è fondamentalmente diversa rispetto alle precedenti, infatti a differenza delle altre due che servono solamente per effettuare test e consentono di lavorare senza protezioni a stretto contatto col robot per via delle molteplici misure di sicurezza, in questo caso è consigliabile stare a distanza e utilizzare protezioni per separare operatore e robot. Questo perché i programmi possono essere eseguiti anche da dispositivi differenti alla teaching pendant

e quindi non sarà presente la funzione di sicurezza del deadman switch, inoltre la velocità come nel caso di T2 non sarà limitata in nessun modo e la modalità di jogging non si potrà usare.

Capitolo 2

Funzioni Collaborative Base

Come già detto nel capitolo precedente, la coesistenza di umani e robot rende più efficiente la produttività nella maggior parte dei casi, però per lavorare assieme nella stessa postazione devono esserci determinate misure di sicurezza per assicurare che l'operatore umano non si faccia male. Proprio per questo sono nati i robot collaborativi, che dispongono di una serie di funzioni che soddisfano determinati requisiti inerenti alla sicurezza degli umani.

2.1 Set Up

Prima di cominciare ad utilizzare le funzioni collaborative però, ci sono alcuni passaggi obbligatori che devono essere fatti. Per consentire una piena comprensione di questa sezione è anche doveroso dare alcune spiegazioni sulla suddivisione delle attività che ogni persona che entra in contatto col robot può svolgere:

- Operatore: Colui che accende e spegne il robot ed esegue i programmi dal pannello di controllo
- Operatore didattico o programmatore: Aziona il robot e gli insegna all'interno dello spazio di lavoro protetto.
- Manutentore tecnico: Svolge gli stessi compiti dell'operatore didattico con l'aggiunta della manutenzione (riparazioni, sostituzione di componenti, regolazioni).

Dalle definizioni precedenti si può intuire come solo il personale formato possa interfacciarsi con il robot nella zona di lavoro protetto, quindi entrarci effettivamente in contatto. Pur essendo formato e

cauto però, gli imprevisti possono sempre accadere. Per questo motivo è presente un sistema chiamato DCS (Dual Check Safety) che si occupa delle fermate di emergenza del robot se si verificano determinate situazioni, in genere legate a velocità e posizione del robot fuori dalle linee guida definite durante il set up o dai limiti imposti dalla casa produttrice.
I parametri del DCS che serviranno alle funzioni del DCS sono immagazzinati in un'area di memoria differente rispetto a quelli delle altre funzioni standard, questo perché l'utilizzatore non può cambiare direttamente i parametri del DCS, essendo fondamentali per la sicurezza come detto precedentemente. Per modificarli l'utilizzatore deve cambiare prima quelli normali nella memoria accessibile a tutti e poi copiarli dalla memoria comune a quella del DCS, questa operazione è detta "Applicazione ai parametri DCS" ed è necessaria una password. Una volta cambiato il parametro di interesse nel DCS, si modificherà automaticamente anche quello standard, inoltre per evitare che ci si dimentichi di aver solo modificato e non applicato i parametri, una volta usciti dalla sezione apposita per la modifica dei parametri standard, sarà segnalato un errore che non scomparirà finché non saranno anche applicate o annullate le modifiche.
I passaggi obbligatori da effettuare durante il set up possono essere riassunti nell'elenco sottostante:

- Configurare i vari limiti per la funzione di contact stop motion: Questa operazione serve ad impostare fino a quattro limiti di forza esterna massima che il robot può sopportare prima di arrestarsi. Si potrà selezionare solo un limite alla volta e dovrà essere modificato con la procedura del DCS spiegata precedentemente. Come si può vedere in figura 2.1 il simbolo

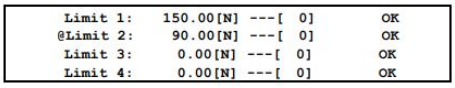

Figura 2.1: Menù dei limiti delle forze esterne

  @ indica quale dei limiti di forza è attualmente selezionato, inoltre se il limite è impostato su 0 N significa che quello è disabilitato.

- Impostare i vari parametri del payload, ovvero la portata del polso, è il massimo peso degli oggetti che il robot può trasportare. E' importante ricordare che in questo valore deve essere

compreso anche il peso del tool attaccato al polso del robot che in alcuni casi potrebbe influenzare non poco, inoltre bisogna specificare anche il centro di massa per avere la miglior accuratezza possibile. Altro parametro molto importante è il payload error margin, ovvero di quanto peso è consentito sforare rispetto al payload impostato, nel momento in cui si supera questo limite, il robot si fermerà automaticamente. Inoltre è possibile abilitare il paylode change distance, una funzione che consente di avere abilitata o disabilitata la contact stop motion in base alla distanza e alla rotazione del polso del robot. Questa può essere impostata per ogni asse X,Y,Z e per una rotazione generale, senza nessun riferimento. Tutto questo è possibile farlo per più payload salvando fino a 10 set up predefiniti tra i quali scegliere una volta acceso il robot, magari per diversi tool utilizzati o in base al tipo di applicazione(figura 2.2). Infine ad ogni accensione bisognerà confermare quale dei payload salvati corrisponde a quello che si andrà ad utilizzare in quella sessione di lavoro, per fare questo sarà necessario sapere anche il code number (master).

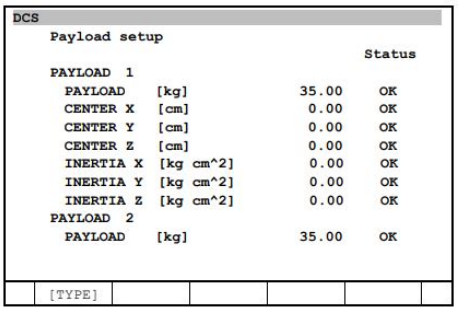

Figura 2.2: Menù dei set up predefiniti del payload

- Settare le velocità massime per le varie modalità di funzionamento T1,T2 e auto. Se durante il funzionamento verranno superati questi limiti, il cobot si fermerà automaticamente, a meno che non si attivi la modalità di Speed Clamping, ovvero una funzione che consente di diminuire la velocità al valore limite ogni volta che si supera.

2.2 Funzioni collaborative

Dopo aver completato il set up iniziale si può cominciare ad usufruire delle varie funzioni collaborative legate in particolare alla sicurezza, queste si differenziano principalmente in strategie pre-collisione e post-collisione. Le prime servono ad anticipare ed evitare le collisioni, hanno bisogno di particolari sensori montati sul robot per determinare se una persona o un oggetto è nelle sue vicinanze. Le seconde invece sono quelle che verranno brevemente trattate di seguito e vengono attivate una volta che la collisione avviene e si basano prevalentemente sul limitare potenza o velocità del robot. Bisognerà abilitarle dal menù legato appunto alla parte collaborativa del robot. Infatti queste funzioni si attiveranno tramite segnali output digitali del robot che si andranno a modificare in base alle condizioni soddisfatte della specifica funzione come per esempio il limite di forze esterne superato della contact stop motion.

2.2.1 Contact stop function

Come anticipato nella sezione precedente, questa funzione si attiva quando la forza esterna supera il limite impostato inizialmente, in questo caso il robot e il programma in esecuzione si arresteranno. Una volta fermo, non potrà tornare a muoversi se la forza non tornerà sotto il valore consentito. Nel caso in cui siano eseguiti più programmi contemporaneamente il robot si arresterà comunque interrompendo anche l'esecuzione dei programmi in parallelo. Però nel caso si voglia che il robot continui comunque ad eseguire un programma indipendentemente dalla contact stop motion, è possibile abilitare una proprietà, chiamata Interruption Disable, che consentirà al programma di continuare a funzionare.
Questa funzione può essere abilitata anche durante la modalità T1 e T2 della teaching mode, le informazioni date precedentemente valgono in ugual maniera anche per questo caso.
Nel caso la forza esterna sia causata dal contatto del robot contro una superficie fissa, può capitare che la forza sia vista dal sensore come continua e di conseguenza il robot non potrà tornare a muoversi se non eseguendo una delle due procedure descritte di seguito:

- Utilizzare la funzione jog direttamente dalla teaching pendant per muovere il robot nella direzione opposta rispetto quella in cui stava andando prima del contatto. In questo modo la forza

esterna non sarà più percepita e dopo aver premuto il tasto di reset si potrà continuare con l'esecuzione del programma.

- Premere contemporaneamente i tasti SHIFT e reset, in questo modo si disabiliterà la funzione di contact stop e si potrà spostare il robot.

2.2.2 Retreat after contact stop

Questa è una variante della contact stop motion, si può utilizzare solo durante l'esecuzione di un programma, quindi non è possibile usufruirne nella modalità di jogging, ovvero quando si fa muovere il robot direttamente dai comandi della teaching pendant. Grazie a questa funzione, è possibile risolvere una problematica descritta in precedenza, ovvero quando il robot entra in contatto con una superficie inamovibile restando bloccato poiché il sensore continua a vedere una forza esterna. Se si dovesse verificare una situazione simile mentre questa funzione è abilitata durante l'esecuzione di un programma, il robot dopo essersi fermato si muoverà autonomamente nella direzione opposta per un tempo e con una velocità impostati.

2.2.3 Push to escape

Un'altra funzione utilizzata per la sicurezza dell'operatore o l'ambiente circostante, può essere usata solamente in auto mode, la teaching pendand deve essere spenta e dopo averla utilizzata il robot si ferma come se fosse stata attivata la contact stop motion. Durante l'esecuzione di un programma se l'operatore spinge il robot, il giunto impostato J1 o J2 si muoverà solamente nella direzione della spinta (figura 2.3) per poi fermarsi poco dopo averla ricevuta. Questa funzione serve per evitare che il robot vada a sbattere contro qualcuno o qualcosa mentre sta eseguendo una task. L'unica cosa a cui prestare attenzione è che la forza della spinta deve essere inferiore al limite di forza settato inizialmente, infatti questa funzione può essere eseguita solo quando è abilitata anche la contact stop motion, quindi nel caso la forza della spinta dovesse essere superiore al valore limite, il robot si fermerà e basta. Alcuni consigli per utilizzare questa funzione al meglio sono di esercitare la forza di spinta lontano dagli assi di rotazione che si vogliono far ruotare e nel caso il robot non si

muovesse dopo una prima spinta, staccare la mano e riprovare con una forza inferiore.

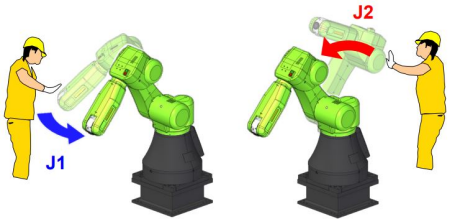

Figura 2.3: Movimento dei giunti J1 e J2 nel push to escape

2.2.4 Manual Guided Teaching

Al contrario delle altre, questa non fa parte delle strategie di pre-collisione o post-collisone legate alla sicurezza. E' una delle funzioni più utili del robot collaborativo, garantisce all'operatore di afferrare il polso del robot e muoverlo a suo piacimento anche con il tool alla fine del braccio, nei limiti imposti dei giunti. Questa funzione risulta di grande aiuto anche per i collaboratori umani che non hanno nozioni di programmazione e che quindi potrebbero utilizzarla per far imparare al robot a seguire uno specifico percorso registrando vari punti e salvandoli nei registri di posizione o arrivare a ridosso di un oggetto da prendere senza dover scrivere del codice. Bisogna sempre ricordarsi di non applicare mai una forza troppo grande al polso altrimenti si attiverà il contact stop motion che farà arrestare il robot. Possono esserci tre principali modi per utilizzarlo:

- Libero: Si può operare in tutti i gradi di libertà, quindi sono consentite sia traslazioni che rotazioni. Purtroppo questo porta spesso ad avere delle imprecisioni in quanto se per esempio si vuole effettuare una pura traslazione applicando una forza in una determinata direzione, il sensore di forza percepirà anche un momento lungo un differente asse di rotazione, questo porta ad un movimento di rototraslazione indesiderato.

- Traslazione: In questa modalità saranno consentite solo traslazioni, sia positive che negative, lungo i tre assi principali X, Y e Z. Si otterrà quindi lo spostamento in ogni direzione del polso del robot ma mantenendo sempre la stessa orientazione.

- Rotazione: In questo caso saranno consentite solo le rotazioni del polso attorno gli assi principali X,Y e Z. Al contrario della modalità precedente, il polso del robot resterà fermo nello stesso punto ma si potrà modificare liberamente la sua orientazione muovendo il resto del robot.

Questa funzione è quindi una grande risorsa soprattutto per la flessibilità che può fornire al robot e all'operatore per rispondere alle varie necessità date da diversi compiti che possono essere richiesti in un'industria. Purtroppo però, questa funzione è opzionale e non tutti i robot collaborativi possono usufruirne. Quindi si ha cominciato a pensare ad un modo per implementarla al Fanuc CR15-iA del laboratorio in cui è stata svolta questa tesi. Nel seguente capitolo infatti si parlerà dei metodi di controllo che possono essere utilizzati per implementare tramite programmazione la funzione Manual guided teaching.

Capitolo 3

Strategie di Controllo

Fondamentale per il manual guided teaching è l'interazione fisica tra operatore umano e robot, è solo grazie a questa che il robot può rilevare le forze generate dall'uomo e capire come muoversi di conseguenza. Ci sono vari metodi per eseguire un controllo di questo tipo, dal un semplice controllo di forza proporzionale ai sofisticati controllo di ammettenza o controllo di impedenza. Nel primo caso si avrà un controllo molto semplice da realizzare ma al contrario del secondo si perderà molto in trasparenza, ovvero la capacità del controllore di compensare gli errori hardware dovuti a fenomeni come inerzia, attrito, vibrazioni in modo tale da renderli impercettibili dall'utilizzatore. Solitamente la si ottiene tramite un feedback di forza, sempre stando attenti a non avere un ambiente troppo rigido o un'impedenza dell'ambiente sconosciuta altrimenti si potrebbe tendere all'instabilità. Per questo elaborato sono stati sviluppati tre tipi di controllo che si andranno ad approfondire di seguito:

- Controllo proporzionale
- Controllo di ammettenza con modello di oscillatore semplice smorzato
- Controllo mixed proporzionale e di ammettenza

Come già accennato, il primo controllo è più adatto ai piccoli spostamenti in quanto più preciso, invece gli altri due troveranno applicazione prevalentemente nei macro spostamenti.
E' inoltre doveroso far notare come non avendo accesso ad un pieno controllo sulla velocità del robot come sarebbe normalmente richiesto, in questi test sui vari controlli le equazioni normalmente

utilizzate per la velocità sono state usate per ricavare i punti di arrivo del robot, che dovendo essere raggiunti sempre nello stesso tempo, costringeva il robot a muoversi più o meno velocemente in base al punto di arrivo che sarebbe stato identificato dalle equazioni. In questo modo è stato aggirato il problema relativo al non poter impostare una velocità variabile e accurata senza l'aiuto del TP.

3.1 Stabilità

Uno degli obiettivi dello sviluppo del manual guided teaching è il raggiungimento della stabilità, questo però ha bisogno di alcuni chiarimenti in quanto esistono diversi tipi di stabilità desiderata. Quando un sistema è progettato per avere interazioni fisiche con l'operatore umano o l'ambiente esterno possono esserci generalmente 3 tipi di stabilità che poi possono essere esaminati con diverse sfaccettature:

- Stabilità disaccoppiata: Quando il dispositivo non ha interazioni fisiche con altri soggetti o ambienti, quindi è libero da ogni contatto.
- Stabilità in transizione: Quando dalla stabilità disaccoppiata si passa alla stabilità accoppiata o viceversa, ovvero il momento in cui un operatore o l'ambiente entrano in contatto con il robot o si disaccoppiano da esso.
- Stabilità accoppiata: Quando il dispositivo rimane in contatto con l'operatore o l'ambiente e scambia con lui forze e momenti.

Nel caso del manual guided teaching si punta ad ottenere tutti e tre i tipi di stabilità ma questo non sarà facile specialmente nella fase di transizione che potrebbe portare una certa instabilità temporanea. Infatti se due sistemi disaccoppiati tra loro sono stabili non è detto che accoppiandosi non diventino instabili. Allo stesso modo non è sicuro che un sistema accoppiato stabile non porti all'instabilità una volta disaccoppiato. Durante la stabilità accoppiata, il robot e l'operatore o l'ambiente si scambiano potenza meccanica, vengono considerati come un unico sistema accoppiato nel quale il robot andrà a presentare un'ammettenza apparente Y_a e l'umano un'impedenza Z_h che andranno a creare un anello chiuso come mostrato in figura 3.1.
In questo caso, la forza F* è quella generata esternamente dall'operatore in maniera volontaria, F_{int} è la forza della dinamica intrinseca

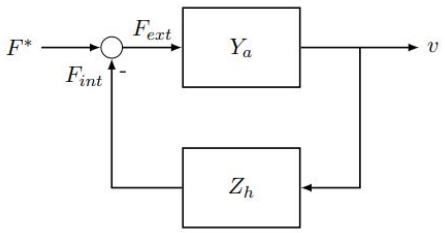

Figura 3.1: Schema a blocchi della stabilità accoppiata tra umano e robot

dell'operatore e sommandole si ottiene la forza F_{ext} effettivamente percepita dal robot che attraversando la sua dinamica del sistema rappresentata da Y_a darà in uscita la velocità v con la quale andrà a muoversi il robot. Il risultato tra questo controllo del dispositivo in ammettenza e l'operatore umano che fa da impedenza chiudendo l'anello di retroazione crea un feedback "negativo", quindi la stabilità del sistema dipende prevalentemente dalle caratteristiche di impedenza dell'utente.
Un'altra informazione molto importante in questo ambito è che l'interazione tra uomo e oggetti energeticamente passivi consente una relazione stabile. Quindi se il robot ha una dinamica apparente energicamente passiva, dovrebbe far si che esso abbia la stabilità accoppiata con l'operatore. Però per essere energeticamente passivo un sistema non deve emettere più energia di quanta ne sia stata immessa, quindi l'integrale nel tempo da meno infinito al tempo presente della moltiplicazione tra la forza e la velocità del robot deve essere sempre maggiore di zero. Nella pratica però imporre la passività nelle interazioni uomo-macchina è troppo conservativo perché sapendo che l'impedenza dell'operatore durante il contatto è molto limitata, se si vuole una stabilità accoppiata con un arto che è di conseguenza considerato infinitamente rigido oppure con una massa inerziale infinita è sicuramente conservativo.

3.2 Controllo Proporzionale

Il metodo di controllo più semplice a livello sia computazionale che intuitivo. L'idea di base è far si che la velocità di spostamento, oppure il punto di arrivo pianificato in questo caso, sia direttamente proporzionale alla forza applicata al polso del robot. Quindi la re-

lazione tra forza e velocità che si cerca per questo tipo di controllo è una semplice equazione proporzionale:

$$v = K_p * F_{ext}$$

dove K_p è il guadagno proporzionale scelto in base a vari test empirici.
Avendo però 6 gradi di libertà (3 traslazioni e 3 rotazioni), nel programma bisognerà calcolare ognuna di queste per far si che il robot si muova contemporaneamente in tutti gli assi che servono per arrivare al punto finale. Questo però comporta dei problemi riguardanti la contemporaneità di traslazioni e rotazioni. Infatti come già detto in precedenza, se per esempio si applica al polso solamente una forza orizzontale X, oltre a quella il sensore di forza percepirà anche una coppia lungo l'asse ortogonale al polso e quindi comincerà ad eseguire oltre alla traslazione anche una rotazione. La soluzione proposta che verrà usata per questo e gli altri tipi di controllo è quella di separare completamente la parte di traslazione e la parte di rotazione durante il programma per mezzo di un particolare evento che verrà chiarito durante la spiegazione del programma. In questo modo i programmi saranno divisi in due macro parti che opereranno allo stesso modo per ricavare la posizione o l'angolazione del tool alla quale arrivare ma compiendo solo rotazioni o solo traslazioni in base alla macro parte del programma in cui ci si trova. Solitamente quindi prima ci posiziona il tool nel punto voluto tramite traslazioni e successivamente si modifica la sua angolazione tramite le rotazioni. Oltre a questo durante le prove è emersa un'altra problematica riguardante l'accuratezza del sensore di forza, è quindi stato necessario settare dei limiti di forza sotto i quali il robot non si sarebbe dovuto muovere, in questo modo si sono potute eliminare le incertezze dovute al sensore oppure a posizionamenti del robot nei quali venivano rilevate forze non coincidenti con quelle attese.

3.3 Controllo di ammettenza con modello di oscillatore semplice smorzato

Questo controllo consente di assegnare un comportamento dinamico desiderato dopo un'interazione tra EOAT e ambiente circostante. Le prestazioni saranno definite da un'impedenza dinamica generalizzata composta da equazioni massa, molla e smorzatore. Questo

consente di non richiedere un modello esplicito delle forze derivanti dall'ambiente esterno. E' adatto a rendere il movimento del robot molto fluido ma durante l'applicazione di piccole forze perde in accuratezza.
Per avere un'idea generale del suo funzionamento in figura 3.2 è rappresentato uno schema a blocchi semplificato di un controllo di ammettenza disaccoppiato, ovvero trascurando la dinamica introdotta dal contatto con l'operatore.

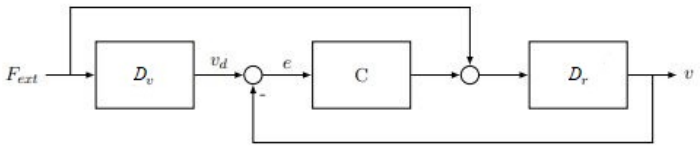

Figura 3.2: Schema a blocchi semplificato di un controllo di ammettenza disaccoppiato

Si può notare come tutto comincia con la forza esterna applicata dall'operatore F_{ext}, questa passa attraverso la dinamica virtuale D_v grazie alla quale genera la velocità di riferimento ideale v_d. Successivamente il controllore C prova ad imporre quella velocità al robot applicando una forza detta Forza di controllo F_c. Contemporaneamente anche la forza F_{ext} agisce sulla dinamica del robot D_r direttamente. Quindi sia F_{ext} sia F_c entreranno nella dinamica del robot dando in uscita un movimento a velocità v. Se non si ha bisogno di limiti particolari da imporre su posizione o velocità, oltre a quelli legati allo spazio di lavoro del robot, allora si cercherà di avere un'ammettenza molto alta e di conseguenza un'impedenza molto bassa per consentire una libertà maggiore all'operatore nel movimento del robot. Ovviamente un'ammettenza infinita è impossibile per la divisione per zero che si verrebbe a creare nella relazione tra forza e velocità, per questo solitamente si utilizza un modello con pura inerzia virtuale più bassa possibile, quel tanto che basta per mantenere la stabilità del sistema. Questo metodo garantisce bassa impedenza per le basse frequenze, attenuazione per le alte e un comportamento non dissipativo. Inoltre minimizza la forza di iterazione tra dispositivo e utilizzatore, così da non farlo stancare in caso di operazioni prolungate nel tempo. L'unica nota negativa è che più l'inerzia virtuale è bassa più si tende all'instabilità, quindi bisognerà trovare il giusto compromesso.

Solitamente il controllo di impedenza con feedback di forza è più utilizzato rispetto quello di ammettenza perché più economico da realizzare, questo però ha uno svantaggio non trascurabile nel tipo di applicazione che si vuole realizzare in questo caso. Nel controllo di impedenza la forza viene controllata solo dopo la misurazione di un movimento o una deviazione dal riferimento, al contrario nel controllo di ammettenza prima viene misurata la forza e solo successivamente si esegue il movimento.
Il secondo metodo, è quindi molto più propenso ad essere usato in catena aperta, risultando quindi facile ed economico da realizzare come già detto in precedenza perché non ha bisogno di costosi sensori aggiuntivi ma anche molto più soggetto alla dinamica disturbante di attriti, specialmente nel caso di corpi rigidi abbastanza pesanti. Quindi in un'applicazione come quella di un braccio antropomorfo non è sicuramente indicata, per questo si predilige il controllo di ammettenza.

3.3.1 Modello oscillatore semplice smorzato

Si procede ora ad ottenere il modello della dinamica virtuale di un oscillatore semplice smorzato. Ovviamente dopo aver trovato il modello corretto in tempo continuo sarà necessario discretizzarlo per poterlo implementare nel programma che andrà a controllare il braccio robotico. Anche in questo caso, il risultato che otterremo dall'equazione andrà applicato a tutti i gradi di libertà del robot per avere una copertura a tutto tondo dei movimenti possibili.
Il punto di partenza è l'equazione monodimensionale di ammettenza:

$$f_H(t) = m_v(\ddot{x}(t) - \ddot{x_0}(t)) + c_v(\dot{x}(t) - \dot{x_0}(t)) + k_v(x(t) - x_0(t))$$

dove:

- $f_H(t)$: Forza di interazione applicata dall'operatore
- m_v: Massa virtuale
- c_v: Costante smorzamento viscoso virtuale
- k_v: Rigidezza virtuale

- $x_0(t)$: Condizione di equilibrio
- $x(t)$,$\dot{x}(t)$,$\ddot{x}(t)$:posizione, velocità e accelerazione.

Se si vuole ottenere uno spostamento libero, bisogna porre a zero $x_0(t)$,$\dot{x}_0(t)$ e $\ddot{x}_0(t)$ ovvero le condizioni iniziali di posizione, velocità e accelerazione, quindi l'equazione di ammettenza con queste condizioni diventa:

$$f_H(t) = m_v\ddot{x}(t) + c_v\dot{x}(t)$$

Successivamente per trovare la velocità desiderata si scrive l'equazione nel dominio di Laplace facendola diventare:

$$\dot{X}_d(s) = V_d(s) = \frac{F_H(s)}{m_v s + c_v} = F_H(s)Y_v(s)$$

dove la velocità sarà la trasformata di laplace di $\dot{x}(t)$ e $Y_v(s)$ come in accennato precedentemente rappresenta la dinamica virtuale di questo sistema.
Si avrà quindi un sistema ad anello aperto che per il momento può essere rappresentato come mostrato in figura 3.3.

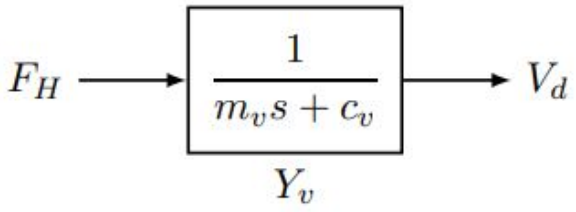

Figura 3.3: Dinamica oscillatore smorzato ad anello aperto

In questo caso la massa virtuale filtrerà il rumore del sensore di forza e smorzerà anche variazioni di forza troppo grandi e improvvise, però se la si imposta ad un valore troppo alto l'interazione con il robot diventa contro intuitiva perché una volta in movimento richiede tempo per fermarsi e cambiare direzione per la troppa inerzia.
Ora si prende come riferimento un oscillatore semplice smorzato con massa m, molla di costante elastica k e smorzatore con costante di smorzamento c come quello mostrato in figura 3.4.

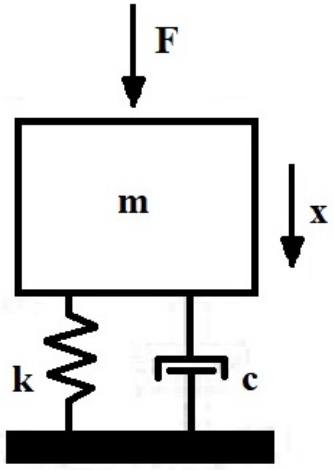

Figura 3.4: Schema oscillatore semplice smorzato

Partendo dalla sua equazione dinamica:

$$F(t) - c\dot{x}(t) - kx(t) = m\ddot{x}(t)$$

si isola la ẍ(t) e si definisce il seguente cambio di variabili

$$\begin{cases} x = x_1 \\ \dot{x} = x_2 \end{cases}$$

Omettendo poi la dipendenza dal tempo e risolvendo il sistema si otterranno queste equazioni:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \dfrac{F - cx_2 - kx_1}{m} \end{cases}$$

Per poter utilizzare queste equazioni nel programma però, è necessario discretizzarle nel tempo, in questo modo si potrà ottenere una soluzione del sistema in modo iterativo, quindi come se ad ogni intervallo di tempo si aggiorneranno le forze e le velocità, o posizioni

nel caso in esame, e con esse anche il movimento del braccio robotico. Si ipotizza un passo k e un periodo di campionamento T_s, e si sostituisce la derivata della posizione con la sua approssimazione in tempo discreto ovvero

$$\dot{x} \cong \frac{x(k+1) - x(k)}{T_s}$$

svolgendo poi il sistema nel tempo discreto e utilizzando l'approssimazione in tempo discreto della derivata si ottiene il seguente risultato:

$$\begin{cases} x_2(k) = \dfrac{x_1(k+1) - x_1(k)}{T_s} \\ \dfrac{x_2(k+1) - x_2(k)}{T_s} = \dfrac{F(k) - cx_2(k) - kx_1(k)}{m} \end{cases}$$

$$\begin{cases} x_1(k+1) = x_1(k) + T_s x_2(k) \\ x_2(k+1) = x_2(k) + T_s \dfrac{F(k) - cx_2(k) - kx_1(k)}{m} \end{cases}$$

$$\begin{cases} x_1(k+1) = x_1(k) + T_s x_2(k) \\ x_2(k+1) = -\dfrac{T_s k}{m} x_1(k) + \left(1 - \dfrac{T_s c}{m}\right) x_2(k) + \dfrac{T_s}{m} F(k) \end{cases}$$

dove $x_1(k+1)$ e $x_2(k+1)$ altro non sono che le prossime posizioni e velocità che verranno usate per l'algoritmo iterativo che continuerà ad aggiornarsi finché l'operatore non smetterà di applicare forze esterne al robot.
Tra le variabili delle equazioni finali alcune dovranno essere ricavate per via empirica, quindi una volta trovati i valori di massa virtuale, costante di smorzamento viscoso virtuale e periodo di campionamento si imposteranno e saranno tenuti costanti durante i vari test. Il compromesso che si cerca di avere è un sistema con un'inerzia non troppo elevata da ottenere un effetto di utilizzo contro intuitivo del manual guided teaching, non troppo bassa per evitare di tendere all'instabilità ma allo stesso tempo percepibile dall'operatore per impedire bruschi e improvvisi cambi di direzione. Per quanto riguarda il coefficiente di smorzamento invece si modifica cercando di

trovare una soddisfacente fluidità di movimento del braccio robotico senza incentivare troppo l'effetto di scivolamento causato se fosse impostato un valore troppo basso. L'unico parametro che cambierà una volta impostati tutti gli altri in base al risultato che vogliamo ottenere e con cui si faranno vari test sarà la rigidezza virtuale k (rappresentata anche dalla costante elastica della molla dell'oscillatore). Ponendola inizialmente a zero durante il bilanciamento degli altri parametri e successivamente aumentandola per far si che il sistema risulti più rigido e che tenda a tornare leggermente verso il punto di partenza dopo uno spostamento.

3.4 Controllo Mixed Proporzionale e di Ammettenza

In precedenza sono stati spiegati i vantaggi e gli svantaggi dei due tipi di controlli. Come è stato detto infatti, il controllo proporzionale manca di fluidità ma quando si applicano piccole forze è estremamente preciso, questo perché per com'è stato scritto il programma, il coefficiente di proporzionalità è legato in maniera direttamente proporzionale alle forze applicate, quindi se applico una piccola forza si sposterà di poco, garantendo così un buon grado di precisione. Al contrario invece, il controllo di ammettenza con modello di oscillatore semplice smorzato tiene conto di molte più informazioni come inerzia, smorzamento e rigidezza, consentendo così una fluidità di movimento molto buona. Però durante i piccoli spostamenti, l'inerzia e la rigidezza vanno a disturbare e sporcare la precisione del movimento, risultando a volte contro intuitive per l'operatore.
Da qui l'idea di creare un programma che inglobasse entrambi i metodi, cercando di enfatizzare solo i vantaggi di ognuno dei due. Quindi tenendo la stessa struttura è stato scritto questo tipo di controllo nel quale sopra una certa soglia di forza applicata sarà utilizzato il controllo di ammettenza con modello oscillatore smorzato, invece sotto quella soglia sarà utilizzato il controllo proporzionale. In questo modo si avrà fluidità durante i grandi spostamenti e precisione nei piccoli. L'unico problema che ci si aspetta da questo tipo di controllo sarà durante le transizioni tra le due soglie, infatti passando da un controllo molto fluido ad uno poco scorrevole e viceversa, la transizione può risultare poco piacevole e sorgente di imprecisioni

di movimento. Per questo, bisognerà trovare il giusto bilanciamento tra una soglia abbastanza bassa da rendere poco percettibile il passaggio tra i due tipi di controllo ma abbastanza alto da permettere durante i valori più bassi di forza del controllo di ammettenza di rendere il movimento contro intuitivo per colpa dell'inerzia e gli altri fattori.

Capitolo 4

Programmi Implementati di Manual Guided Teaching

Si passa ora all'implementazione dei programmi sviluppati con i diversi metodi di controllo.
Per scrivere il programma è stato usato il linguaggio di programmazione Karel di Fanuc. Inizialmente era utilizzato solamente per insegnare le basi della programmazione robotica a studenti o principianti per poi diventare il principale linguaggio di programmazione di robot e controllori Fanuc.

4.1 Basi del linguaggio Karel di Fanuc

Prima di cominciare a spiegare nel dettaglio il codice, è doveroso dare una breve delucidazione sugli elementi fondamentali e le principali differenze che il Karel di Fanuc ha rispetto ai linguaggi più comuni. Prima di tutto bisogna specificare che in questo ambiente non sarà possibile utilizzare o creare variabili come nella grande maggioranza degli altri linguaggi, in questo caso quindi si farà uso dei registri. Nel programma si utilizzeranno principalmente due tipi di registri:

- Registri numerici R[n]: Sono variabili numeriche che possono contenere un numero intero o reale. Solitamente nel sistema sono presenti un minimo di 200 registri. La lettera n all'interno delle parentesi quadre specifica il numero del registro.

- Registri di posizione PR[n]: Sono variabili che contengono una specifica posizione data da sei elementi (x,y,z,w,p,r), le prime

tre rappresentano una posizione nello spazio e le altre l'angolazione su ogni asse.E' possibile utilizzare sia tutto il registro di posizione sia uno solo dei suoi campi.

Per pura nota informativa si precisa che esiste anche un terzo tipo di registro, chiamato Registro di visione VR, ma si occupa di comunicare con il sistema di visione Fanuc iRVision e quindi non avendo utilità per l'obiettivo di questo elaborato non è stato utilizzato nel codice.
Altra peculiarità di questo linguaggio sono le etichette, ovvero delle istruzioni che consentono di definire una destinazione di una diramazione, ovvero una condizione che consente di spostarsi da una parte all'altra del programma tramite l'istruzione Jump LBL. In questo modo, sfruttando Jump e etichette si possono creare dei loop infiniti, come i cicli while nel linguaggio C, molto utili per sfruttare la continuità di aggiornamento delle forze e degli spostamenti del braccio meccanico.
Ultima grande differenza con i linguaggi di programmazione più comuni è l'istruzione di movimento del robot. Ci sono infatti quattro diversi modi per arrivare nella posizione desiderata tramite interpolazione: joint, lineare, circolare e circle arc.
Al contrario delle altre tre, l'interpolazione joint non tiene conto della traiettoria e l'orientamento del tool e volendo quindi uno spostamento meno condizionato possibile per consentire maggiore libertà di movimento all'operatore che guida il robot, si utilizzerà solo questo metodo. Con l'interpolazione joint il robot si muove su tutti gli assi allo stesso tempo, coordinando i vari motori con diverse velocità in modo tale da arrivare al punto desiderato contemporaneamente e fermare tutti i motori. La velocità è espressa in percentuale rispetto alla massima consentita ed è calcolata per ogni asse. Grazie a questa interpolazione è inoltre possibile specificare come terminare l'istruzione di movimento tramite le sigle FINE e CNT.
FINE serve per assicurarsi che in una serie di istruzioni di movimento, il robot arrivi nella destinazione prefissata, si fermi e solo successivamente prosegua verso la prossima. Al contrario con CNT il robot si avvicina alla destinazione ma invece di fermarsi prosegue direttamente verso il punto successivo. Ovviamente all'ultima destinazione designata arriva al punto desiderato e poi si ferma. Nel CNT è possibile impostare un valore da 0 a 100 che sarà direttamente proporzionale a quanto il robot passerà distante dalla destinazione prima di dirigersi verso la successiva. Nei programmi del manual

guided teaching è stato utilizzato il CNT100 per garantire una maggiore fluidità di movimento e una minore propensione ai movimenti bruschi.
Il resto delle informazioni verrà dato durante la spiegazione dei programmi dei vari metodi di controllo.

4.2 Controllo Proporzionale

La logica alla base di questo programma è tenere un registro (R[81]) che impone di utilizzare la traslazione o la rotazione in base al valore 1 o 2 che può assumere. Ad inizio programma quindi, oltre a definire quale tool stiamo utilizzando, si impone il valore 1 al registro 81, così facendo si è sempre sicuri di partire dallo stato di traslazione. Di seguito le prime righe di codice del controllo proporzionale.

```
1  UTOOL_NUM=4
2
3  R[81]=1
4
5  LBL[1]
```

Nella prima riga, come già accennato, si specifica quale dei set già impostati in precedenza utilizzare, il numero 4 corrisponde alla pinza schunk. Invece l'etichetta LBL[1] servirà a far tornare l'esecuzione del programma all'inizio per ricalcolare le nuove forze e i nuovi spostamenti da fare.
Successivamente si procede con l'acquisizione dei valori rilevati dal sensore di forza nei registri dal 33 al 38.

```
7  !Legge le forze e i momenti dal sensore
8  R[33]=$CCC_GRP[1].$UT_FORCE[1]
9  R[34]=$CCC_GRP[1].$UT_FORCE[2]
10 R[35]=$CCC_GRP[1].$UT_FORCE[3]
11 R[36]=$CCC_GRP[1].$UT_FORCE[4]
12 R[37]=$CCC_GRP[1].$UT_FORCE[5]
13 R[38]=$CCC_GRP[1].$UT_FORCE[6]
```

Come già detto nel capitolo 3, il sensore di forza ha un'incertezza che varia anche in base all'asse e alla velocità dei movimenti che il braccio robotico compie. Quindi dopo una serie di testi si sono ricavati dei dati empirici e tramite dei comandi IF ed ELSE è stato impostata una condizione per la quale se anche solo una delle forze rilevate superi i limiti negativi o positivi dell'incertezza, si salta all'etichetta LBL[5] che porterà poi al calcolo della posizione a cui arrivare e al successivo spostamento, in caso contrario si arriverà

all'istruzione di salto JUMP all'etichetta LBL[6] posizionata a fine del programma saltando tutti i calcoli e le istruzioni di movimento, seguita subito dopo da un altro Jump all'etichetta LBL[1] che farà ricominciare il loop del programma.

```
15  !Se forze e momenti sono superioriori alla soglia esegue il movimento
16
17  IF (((R[33]>3)) OR ((R[34]>3)) OR ((R[35]>11)) OR (R[36]>2) OR (R[37]>2) OR (R[38]>2)),JMP LBL[5]
18
19  IF (((R[33]<(-3))) OR ((R[34]<(-3))) OR ((R[35]<(-11))) OR (R[36]<(-2)) OR (R[37]<(-2)) OR (R[38]<(-2))),JMP LBL[5]
20
21  JMP LBL[6]
22
23
24  LBL[5]
```

Una volta passata l'etichetta LBL[5], tramite due brevi righe di codice, si imposta il registro di posizione 31 come quella attuale e nel registro di posizione 32 si impostano tutti i valori dei campi a zero.

```
26    PR[31]=LPOS
27    PR[32]=LPOS-LPOS
```

Successivamente il programma controllerà se il valore del registro R[81] è 1 o 2, in base a quello si eseguiranno i calcoli delle traslazioni o delle rotazioni da fare. Per cambiare il valore del registro 81 da 1 a 2 e viceversa, l'idea è quella di applicare una coppia negativa all'asse Z del braccio, e quindi una rotazione del tool, abbastanza grande (in questo caso maggiore di 3Nm) in modo tale che non sia confusa con una normale coppia che si utilizzerebbe per ruotarlo solitamente. Appena il programma rileva l'applicazione di questa coppia, procederà a cambiare il valore del registro e riportare all'inizio l'esecuzione del codice. Di seguito si riporta il cambio di valore da 1 a 2, nell'altro caso le istruzioni sono le medesime ma si imposta il valore da 2 a 1 e si entra nel primo IF solo se R[81]=2.

```
30  !Se Mz supera un certo valore si cambia da forze assiali a momenti
31  IF (R[81]=1) THEN
32
33  IF (R[38]<(-3)) THEN
34  R[81]=2
35  WAIT   1.00(sec)
36  JMP LBL[1]
37  ENDIF
```

Si procede ora al calcolo della posizione o rotazione da eseguire. In questo tipo di controllo si è scelto di scaglionare il valore della forza che può essere applicata al polso e assegnare un diverso coefficiente di proporzionalità in base all'intensità della forza. Nel seguente screen si riportano gli scaglioni più bassi nei quali i coefficienti di

proporzionalità K_p che moltiplicano le forze acquisite trasformandole così in un valore che rappresenta la distanza o rotazione per ogni asse in millimetri o gradi da percorrere, e che verrà assegnato al corrispondente campo nel registro di posizione PR[32]. Una volta aggiornati correttamente i valori, si salta all'etichetta LBL[7], posizionata prima dell'istruzione di movimento. Di seguito lo screen della parte riguardante le traslazioni, quella per le rotazioni sarà uguale con l'unica differenza riguardante i valori delle soglie per i diversi K_p e i campi di PR[32] nei quali verranno salvati i valori.

```
39  !Se le forze sono superiori ad una certa soglia aumenta la velocità di movimento
40  IF (((R[33]>15)) OR ((R[34]>15)) OR ((R[35]>15))) THEN
41
42  PR[32,1]=R[33]*3
43  PR[32,2]=R[34]*3
44  PR[32,3]=R[35]*3
45
46  JMP LBL[7]
47  ENDIF
48  IF (((R[33]<(-15))) OR ((R[34]<(-15))) OR ((R[35]<(-15)))) THEN
49  PR[32,1]=R[33]*3
50  PR[32,2]=R[34]*3
51  PR[32,3]=R[35]*3
52
53  JMP LBL[7]
54  ENDIF
55
56
57  IF (((R[33]<15)) OR ((R[34]<15)) OR ((R[35]<15))) THEN
58
59  PR[32,1]=R[33]*1
60  PR[32,2]=R[34]*1
61  PR[32,3]=R[35]*1
62
63  JMP LBL[7]
64  ENDIF
65
66  IF (((R[33]>(-15))) OR ((R[34]>(-15))) OR ((R[35]>(-15)))) THEN
67  PR[32,1]=R[33]*1
68  PR[32,2]=R[34]*1
69  PR[32,3]=R[35]*1
70
71  JMP LBL[7]
72  ENDIF
73
74  LBL[7]
75  ENDIF
```

L'ultima cosa che rimane da fare è solamente l'istruzione di spostamento verso PR[32] tramite interpolazione joint e avvicinamento al punto finale con CNT100 prima di ricominciare nuovamente il loop.

```
125   J PR[31] 100% CNT100 Tool_Offset,PR[32]
```

Si torna poi all'inizio del programma tramite l'istruzione di Jump all'etichetta iniziale LBL[1], continuando così a muovere il robot aggiornando ad ogni ciclo la destinazione.

4.3 Controllo di ammettenza modello oscillatore semplice smorzato

La struttura generale di questo programma è simile a quella del controllo proporzionale, la differenza sta nel modo di calcolare la nuova destinazione e il costante ma necessario reset dei vari parametri nel momento in cui l'operatore non applica più la forza sul polso del robot.
Prima di tutto è necessario spiegare a quale dato corrispondono i vari registri inizializzati all'avvio del programma.

```
1  UTOOL_NUM=4
2  !forze assi discretizzate precedenti
3  R[83]=0
4  R[84]=0
5  R[85]=0
6  !Velocità assi discretizzate precedenti
7  R[104]=0
8  R[105]=0
9  R[106]=0
10 !Velocità  tempo reale
11 R[91]=0
12 R[92]=0
13 R[93]=0
14 !Tempo discretizzazione
15 R[82]=.3
16 !Controllo forza impulsiva
17 R[99]=0
18 !Massa virtuale
19 R[101]=65
20 !Rigidezza kv
21 R[102]=20
22 !Smorzamento cv
23 R[103]=100
24 !Posizione attuale
25 PR[31]=LPOS
26 PR[32]=LPOS-LPOS
27
28 LBL[1]
```

Si può notare come il numero di dati rispetto al semplice controllo proporzionale aumenta di molto, nello screen soprastante possiamo identificare:

- R[83],R[84],R[85]:Valori delle forze applicate nel ciclo precedente.
- R[104],R[105],R[106]:Valori delle velocità calcolate nel ciclo precedente.

- R[91],R[92],R[93]: Valori delle velocità calcolate nel ciclo corrente.
- R[82]:Tempo di discretizzazione trovato con metodo empirico
- R[101]:Massa Virtuale del sistema m_v trovato con metodo empirico
- R[102]:Rigidezza del sistema k_v trovato con metodo empirico
- R[103]:Smorzamento del sistema c_v trovato con metodo empirico
- R[99]: Registro che controlla e annulla le forze impulsive indesiderate
- PR[31],PR[32]: Hanno la stessa funzione che avevano nel controllo proporzionale.

Subito dopo l'inizializzazione dei registri c'è l'etichetta LBL[1] che sarà la destinazione di arrivo una volta finito il ciclo per farlo ricominciare.
Si acquisiscono poi i valori delle forze dal sensore nei registri appositi e si verifica che non siano nel range di incertezza come fatto nel programma precedente ma in questo caso è necessaria un'ulteriore operazione. Se le forze rientrano tutte nel range di incertezza, prima di tornare all'inizio del programma sarà necessario resettare i valori come se si fosse all'inizio del programma.

```
41  !Resetto i valori se si ferma
42  R[86]=0
43  R[83]=0
44  R[84]=0
45  R[85]=0
46  R[91]=0
47  R[92]=0
48  R[93]=0
49  R[99]=0
50  R[104]=0
51  R[105]=0
52  R[106]=0
53  JMP LBL[6]
```

Questo perché i registri che contengono le velocità o le forze del ciclo precedente dopo che il robot si è fermato perché l'operatore ha interrotto il contatto devono tornare nulle, altrimenti nel momento in cui

si ricominciasse ad applicare una forza le nuove velocità verrebbero calcolate con valori molto alti che normalmente richiederebbero un po' di tempo per arrivarci e ne scaturirebbe un movimento estremamente brusco e involontario del robot. Per questo si resettano alcuni di questi dati prima di eseguire l'istruzione di Jump e tornare all'inizio del programma come mostrato di seguito.
Se invece si supera la condizione di IF per l'incertezza del sensore, si procede prima con il calcolo della destinazione usando i valori delle velocità precedenti. In questo caso basta la moltiplicazione tra il tempo di discretizzazione e le velocità che aumenteranno o diminuiranno progressivamente in base all'intensità della forza applicata. Per evitare ripetizioni anche in questo caso verranno mostrati i passaggi per le traslazioni, nel caso delle rotazioni il processo sarà il medesimo.

```
75  !Calcolo posizioni X Y Z
76  PR[31]=LPOS
77  PR[32]=LPOS-LPOS
78  PR[32,1]=R[91]*R[82]
79  PR[32,2]=R[92]*R[82]
80  PR[32,3]=R[93]*R[82]
```

Dopo questo calcolo, si sfruttano le equazioni ricavate alla fine del capitolo 3 per il controllo di ammettenza e si ricavano i valori delle nuove velocità. Sarà però necessario effettuare un po' di conversioni perché bisogna ottenere il risultato in mm/s. Di seguito le equazioni utilizzate nel programma.

```
81
82  !Calcolo velocit  X Y Z
83
84  R[91]=((((R[104]/1000)-R[82]*R[103]*(R[104]/1000)/R[101])+(R[82]*R[83]/R[101])-(R[82]*(R[102]/1000)*PR[32,1]/R[101]))*1000)
85
86  R[92]=((((R[105]/1000)-R[82]*R[103]*(R[105]/1000)/R[101])+(R[82]*R[84]/R[101])-(R[82]*(R[102]/1000)*PR[32,2]/R[101]))*1000)
87
88  R[93]=((((R[106]/1000)-R[82]*R[103]*(R[106]/1000)/R[101])+(R[82]*R[85]/R[101])-(R[82]*(R[102]/1000)*PR[32,3]/R[101]))*1000)
89
```

Una volta trovate le nuove velocità, si procede con l'aggiornamento di esse e delle forze acquisite in questo ciclo, rimpiazzando i dati presenti nei registri 83,84,85 per le forze e 104,105,106 per le velocità.
Svolto anche questo passaggio, si arriva all'istruzione di movimento sempre joint e al Jump che farà ricominciare dall'inizio il programma, proprio come nel controllo proporzionale.
Oltre alle funzioni necessarie per il buon funzionamento del programma, ne è stata aggiunta una ulteriore. Questa serve per evitare che urti di breve durata ma grande potenza, spesso accidentali, facciano andare il polso lontano dalla destinazione originaria, questa

```
  !Aggiornamento forze precedenti

  R[83]=R[33]
  R[84]=R[34]
  R[85]=R[35]

  !Aggiornamento velocit precedenti

  R[104]=R[91]
  R[105]=R[92]
  R[106]=R[93]


  !Movimento nella nuova posizione

J PR[31] 100% CNT100 Tool_Offset,PR[32]

  LBL[6]

  JMP LBL[1]
```

funzione si attiva solamente se il robot è ferma e una forza superiore ai 15N e con durata inferiore ai 0.3 secondi venga applicata. Ovviamente questi parametri possono essere modificati in base alle varie esigenze e alle caratteristiche della postazione di lavoro.

```
!Non si muove per spinte impulsive
IF (R[99]=0) AND (((R[33]>15) OR (R[34]>15) OR (R[35]>15)) OR (((R[33]<(-15)) OR (R[34]<(-15)) OR (R[35]<(-15)))) THEN
WAIT    .30(sec)
IF (((R[33]>2)) OR ((R[34]>2)) OR ((R[35]>5)) OR (R[36]>2) OR (R[37]>2) OR (R[38]>2)) THEN
R[99]=1
JMP LBL[7]
ENDIF
IF (((R[33]<(-3))) OR ((R[34]<(-3))) OR ((R[35]<(-7))) OR (R[36]<(-4)) OR (R[37]<(-4)) OR (R[38]<(-4))) THEN
R[99]=1
JMP LBL[7]
ELSE
JMP LBL[6]
ENDIF
ENDIF

LBL[7]
R[99]=1
```

Dal breve tratto di programma qui sopra, si nota come il registro R[99] serva per decretare se il robot era inizialmente in movimento durante l'applicazione della forza superiore a 15N o no, questo per evitare che il robot si fermi durante una progressiva accelerazione dovuta al controllo corretto dell'operatore.
Una volta controllato che la spinta superiore al limite sia stata applicata quando il robot era fermo, si controlla che fosse di tipo impulsivo e quindi probabilmente accidentale, in caso contrario si suppone che sia semplicemente l'operatore che voglia spostare velocemente il polso e si lascia quindi procedere con il normale svolgimento del programma. Se invece la forza è applicata per meno di 0.3s, la funzione non farà avanzare il programma alla parte dell'istruzione di movimento e ricomincerà da capo il programma senza far muovere il robot come se nulla fosse successo.

4.4 Controllo Mixed Proporzionale e di Ammettenza

Per quanto riguarda il controllo Mixed non ha una struttura diversa o funzioni particolari rispetto gli altri due programmi. Come già spiegato nel capitolo precedente si tratta dell'unificazione dei due programmi per far risaltare i loro vantaggi.
La struttura rimane quindi la stessa di quella del controllo di ammettenza ma dopo aver acquisito i valori rilevati dal sensore di forza come al solito, si divide il programma principale in due grandi condizioni, nella prima si controlla se le forze sono superiori ad un certo limite, positivo o negativo, come mostrato di seguito.

```
34  !Controllo Ammettenza
35  IF ((R[33]>12) OR (R[34]>12) OR (R[35]>12)),JMP LBL[5]
36
37  IF (((R[33]<(-12))) OR ((R[34]<(-12))) OR ((R[35]<(-12)))),JMP LBL[5]
38
```

Dentro questa condizione IF ci saranno tutti i processi già mostrati nel controllo di ammettenza con modello oscillatore smorzato.
Se invece non entra in questa condizione potrebbe entrare nella seconda parte del programma, ovvero quella che permetterà un controllo proporzionale.

```
92  !Controllo Proporzionale
93
94  IF (((R[33]>3)) OR ((R[34]>3)) OR ((R[35]>5))),JMP LBL[7]
95
96  IF (((R[33]<(-3))) OR ((R[34]<(-3))) OR ((R[35]<(-7)))),JMP LBL[7]
97
```

Nella medesima maniera, in questo caso si utilizzerebbe il controllo proporzionale spiegato all'inizio ma solo con il suo scaglione più basso di coefficiente di proporzionalità.
Nel caso il programma non entrasse in nessuno dei due IF vorrebbe dire che nessuno sta interagendo con il robot e i valori di forza acquisiti rientrerebbero nel range di incertezza del sensore di forza.

Capitolo 5

Risultati sperimentali

Per quanto riguarda la raccolta dati è stato preso in considerazione solamente lo spostamento sull'asse x, in modo tale da semplificare il confronto tra i vari tipi di controllo.
Per raccogliere tutti questi dati è stato necessario ricorrere ad un programma esterno sviluppato in laboratorio da un collega. Grazie a questo è stato possibile ricavare forza e velocità di ogni metodo.
Al fine di rendere questi dati più confrontabili possibile, si è cercato di utilizzare sempre lo stesso pattern di forze lungo l'asse x del polso.
In figura 5.1 di seguito è mostrato il riferimento di forze applicate che sarà preso da esempio per tutti i test, ovviamente bisognerà considerare un certo margine di errore dato dal fattore umano e quindi una ripetibilità non molto efficiente.

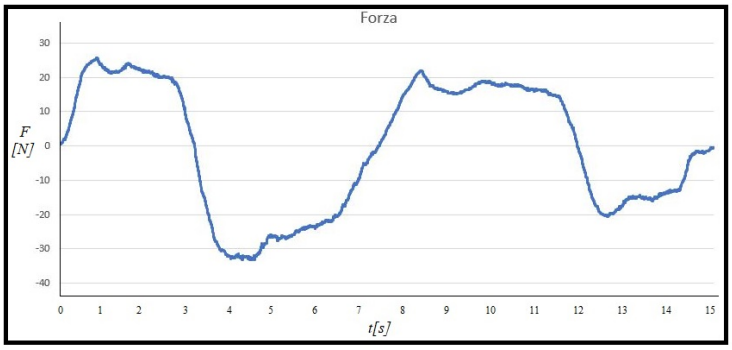

Figura 5.1: Forza usata come riferimento

5.1 Risultati Controllo Proporzionale

Continuando con l'ordine utilizzato anche per i capitoli precedenti, si procede ad analizzare l'andamento delle velocità del controllo proporzionale mostrato in figura 5.2. Come ci si aspettava, la velocità

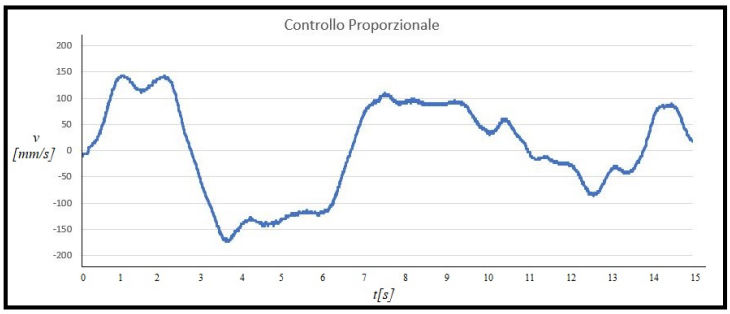

Figura 5.2: Velocità del controllo proporzionale

è abbastanza brusca, durante lo spostamento del robot tramite il polso utilizzando forze abbastanza alte, non si riesce ad avere una buona fluidità di movimento, questo lo si nota anche nel grafico dai continui cambi di pendenza, anche ripidi, della velocità.
Questo metodo di controllo quindi, può essere utile solamente nelle applicazioni dove non è necessaria una grande precisioni nei macro spostamenti e il comfort dato al movimento dell'operatore non è una priorità. Utilizzando poca forza, quindi durante spostamenti piccoli, risulta molto precisa perché la proporzionalità della velocità che dipende dalla forza usata è molto bassa, quindi il robot che si muove lentamente è molto semplice da posizionare dove si vuole con una buona precisione.

5.2 Risultati Controllo di Ammettenza

Si procede ora al controllo di ammettenza con modello di oscillatore semplice smorzato. Prima di tutto, sono state eseguite diverse prove per tarare correttamente il tempo di discretizzazione ideale, la massa virtuale e il coefficiente di smorzamento e si è arrivati a questo risultato finale:

- Tempo di discretizzazione=0.3
- Massa virtuale=65

- Coefficiente d smorzamento=100

Il valore che indica la rigidezza invece, è stato tenuto a zero per permettere una corretta calibrazione delle altre componenti. Il grafico di velocità che si è ottenuto con questi valori, seguendo sempre lo stesso riferimento di forza è mostrato in figura 5.3. Si procede poi

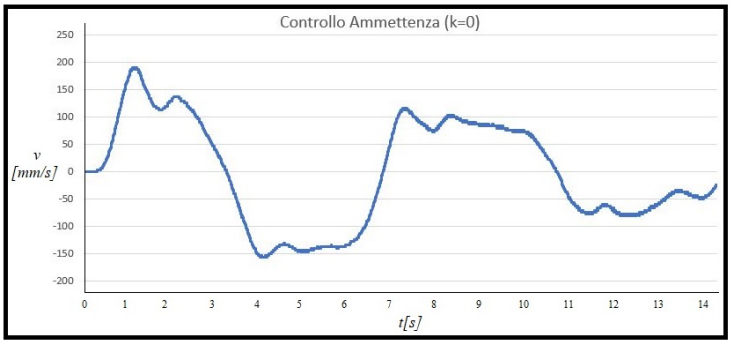

Figura 5.3: Velocità del controllo di ammettenza con rigidezza k=0

con i test empirici per trovare il giusto compromesso per il valore della rigidezza k. Come detto in precedenza, si cerca un compromesso tra una buona fluidità di movimento senza farlo diventare troppo contro intuitivo o difficile da controllare per le componenti negative di bilanciamento che introduce la rigidezza.
Questo compromesso è stato trovato con k=20. Si può notare in figura 5.4 come rispetto al grafico delle velocità con k=0, questo abbia un andamento più dolce e con minori oscillazioni durante i cambi di velocità e accelerazione. Da notare però che durante i piccoli spo-

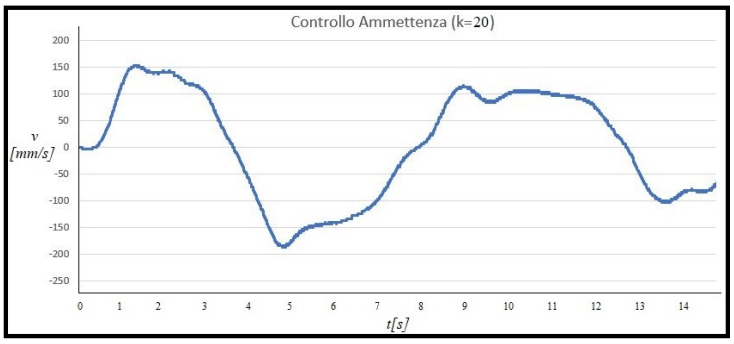

Figura 5.4: Velocità del controllo di ammettenza con rigidezza k=20

stamenti, come ci si aspettava, pur avendo tarato massa virtuale,

rigidezza e coefficiente di smorzamento, è troppo poco preciso, infatti ogni micro spostamento è influenzato dall'inerzia generata dalle tre componenti precedenti.

5.3 Risultati Controllo Mixed

Infine tramite questo tipo controllo ci si aspetta una curva di velocità dolce durante i macro spostamenti dovuta al controllo di ammettenza e una curva sicuramente meno dolce ma che consente maggiore precisione per il posizionamento nei micro spostamenti.

In figura 5.5 si può notare come da previsioni, la differenza di dol-

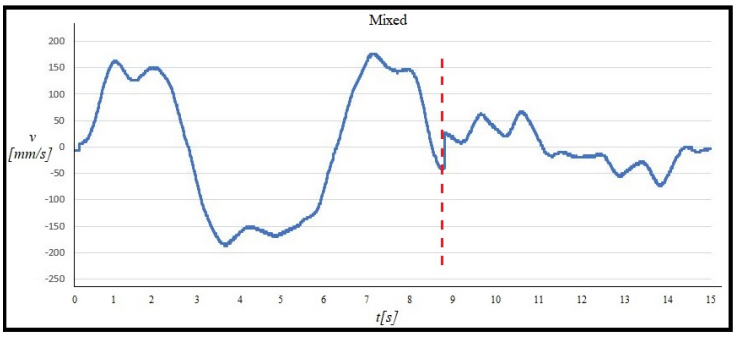

Figura 5.5: Velocità del controllo mixed

cezza tra i due tipi di controlli, divisi dalla linea tratteggiata rossa, che verranno usati per due tipi di movimenti diversi allo scopo di ottenere un movimento complessivo che può soddisfare le esigenze dell'operatore in base alla task che deve essere svolta, che sia seguire un percorso o posizionare precisamente il polso del robot vicino all'oggetto da prendere.

Capitolo 6

Conclusioni

Alla fine di questo elaborato oltre alle funzioni collaborative di base sono stati sperimentati diversi metodi di controllo per il manual guided teaching di un robot che non aveva questa opzione incorporata. I risultati sono stati soddisfacenti in quanto, nonostante i primi due metodi abbiano delle lacune non poco trascurabili, grazie all'implementazione di entrambi nello stesso programma, si è riusciti a sopperire a quelle lacune ed avere come risultato un movimento complessivo soddisfacente, con una curva velocità abbastanza dolce e piacevole da manovrare nei macro spostamenti e un posizionamento preciso del polso nei micro spostamenti. Oltre ad essere di facile utilizzo, una volta avviato il programma non sono necessarie conoscenze informatiche o meccatroniche per manovrare il polso del robot, quindi l'obiettivo di farlo utilizzare anche da personale non formato in quegli ambiti si può ritenere raggiunto. Per il futuro di questa applicazione potrebbe essere implementata una collaborazione ancora più stretta tra cobot e umano, per esempio utilizzare il manual guided teaching anche dopo aver agganciato un eventuale oggetto con il tool a disposizione e trasportarlo dividendosi il peso in una percentuale desiderata e nei limiti del payload con l'operatore. Questo però richiederebbe una pinza a vuoto, in quanto la pinza schunk di cui è attualmente dotato il robot non riesce a trasmettere al sensore tutte le forze subite, specialmente quelle derivate dalle rotazioni, e un migliore bilanciamento riguardante la correzione del peso da sostenere assieme all'operatore per non percepire solamente una forza verso il basso mentre lo si solleva e quindi inevitabilmente ottenere uno spostamento verso la stessa direzione. In conclusione, il lavoro svolto in questo elaborato si può ritenere positivo e con molti margini di miglioramento dati da diversi utilizzi per il futuro.

Bibliografia

[1] G.Lefranca,I.Lopez-Juarezb,R.Osorio-Comparánc,M.Peña-Cabrera. <*Impact of Cobots on automation*>, Procedia Computer Science, 214, 2022, pp.71-78, DOI: https://doi.org/10.1016/j.procs.2022.11.150.

[2] E.Matheson,R.Minto,E.Zampieri,M.Faccio,G.Rosati. <*Human–Robot Collaboration in Manufacturing Applications: A Review*>, Robotics 8(4), 100, DOI: https://doi.org/10.3390/robotics8040100.

[3] J.A.Coletta,V.Chauhan.<*Teaching Industrial Robot Programming Using Fanuc Roboguide and iRVision Software*>,Cybernetics and Informatics,978-1-950492-67-1,pp.45-50, DOI: https://www.iiis.org/CDs2022/CD2022Summer/papers/EA089JN.pdf.

[4] B.Whitsell,P.Artemiadis.<*Physical Human–Robot Interaction (pHRI) in 6 DOF With Asymmetric Cooperation*>,IEEE Access,5,pp.10834-10845, 2017, DOI: 10.1109/ACCESS.2017.2708658.

[5] Fanuc Corporation.<*Collaborative Robot Function:Operator's Manual*>,Fanuc Robot Series, 2015, DOI: B-83744EN/05.

[6] Fanuc Corporation.<*Force Sensor:Operator's Manual*>,Fanuc Robot Series, 2017

[7] M.Safeea,R.Bearee,P.Neto.<*End-Effector Precise Hand-Guiding for Collaborative Robots*>,ROBOT 2017:Third Iberian Robotics Conference,694,2017, DOI: https://doi.org/10.1007/978-3-319-70836-2_49.

[8] R.Ikeura,H.Inooka.<*Variable Impedance Control of a Robot for Cooperation with a Human*>,Proceedings of 1995 IEEE Interna-

tional Conference on Robotics and Automation1995,3,pp.3097-3102, DOI: 10.1109/ROBOT.1995.525725.