

UNIVERSITÀ DEGLI STUDI DI BERGAMO  
Scuola di Ingegneria  
Dipartimento di Ingegneria Gestionale, dell'Informazione  
e della Produzione  
Corso di Laurea Magistrale in Ingegneria Informatica

**Indagine teorica e sperimentale sul  
controllo centralizzato per un  
manipolatore a cinematica parallela**

**Tesi magistrale**

Daniele Ravasio, mat. 1045934

Simone Cortinovis, mat.

**Anno Accademico 2020-2021**



# Indice

<b>1</b>	<b>Introduzione</b>	<b>5</b>
1.1	Obiettivo . . . . .	5
1.2	Stato dell'arte . . . . .	5
1.3	Ambiti applicativi . . . . .	6
1.4	Software modellazione teorica . . . . .	7
1.4.1	Matlab . . . . .	7
1.4.2	Adams . . . . .	7
<b>2</b>	<b>Cinematica Manipolatore</b>	<b>9</b>
2.1	Cinematica Diretta . . . . .	10
2.1.1	Posizione . . . . .	10
2.1.2	Velocità . . . . .	12
2.1.3	Accelerazione . . . . .	13
2.2	Cinematica inversa . . . . .	14
2.2.1	Posizione . . . . .	14
2.2.2	Velocità . . . . .	15
2.2.3	Accelerazione . . . . .	16
<b>3</b>	<b>Dinamica Manipolatore</b>	<b>17</b>
3.1	Prerequisiti per il calcolo della dinamica . . . . .	17
3.1.1	Jacobiana $J_{34}$ e calcolo $\dot{\theta}_3, \dot{\theta}_4$ . . . . .	17
3.1.2	Jacobiana $J_{34}$ e calcolo di $\ddot{\theta}_3, \ddot{\theta}_4$ . . . . .	18
3.1.3	Matrici di inerzia . . . . .	19
3.2	Principio dei lavori virtuali . . . . .	20
3.3	Dinamica inversa . . . . .	20
3.4	Dinamica diretta . . . . .	21
Modellazione su Adams . . . . .	23	
Validazione e confronto . . . . .	24	
3.5	Punti di singolarità . . . . .	25
Primo e secondo caso . . . . .	26	

Terzo caso . . . . .	27
Terzo e quarto caso . . . . .	28
3.6 Manipolabilità . . . . .	29
3.7 Workspace . . . . .	30
<b>4 Modellazione end-effector</b>	<b>33</b>
4.1 Cinematica end-effector . . . . .	34
4.2 Dinamica della vite . . . . .	35
<b>5 Tecnologie implementate</b>	<b>36</b>
5.1 Simulink Real time . . . . .	36
5.2 EtherCAT . . . . .	36
5.2.1 Proprietà . . . . .	37
5.2.2 Gestione della rete . . . . .	37
5.2.3 Implementazione interfacce . . . . .	38
5.3 CME2 . . . . .	38
5.4 EC-Engineer . . . . .	40
<b>6 Sistema reale</b>	<b>41</b>
6.1 Struttura del robot . . . . .	41
6.1.1 Azionamenti . . . . .	42
6.1.2 Beckhoff EK1814 . . . . .	42
6.1.3 Configurazione della rete . . . . .	43
6.2 Implementazione nel sistema reale . . . . .	46
Inizializzazione . . . . .	46
Input . . . . .	47
Stateflow . . . . .	48
Controllo vite . . . . .	49
Controllo braccia . . . . .	50
Coppie uscita . . . . .	50
6.3 Stateflow . . . . .	52
6.3.1 Fase di Homing . . . . .	52

6.3.2	Fase di posizionamento . . . . .	54
6.3.3	Fase di controllo . . . . .	56
	Gestione variabile di stato e luci . . . . .	58
6.3.4	Interfaccia grafica . . . . .	58
6.4	Controllo vite . . . . .	60
6.4.1	Controllo proporzionale . . . . .	60
6.4.2	Controllo proporzionale derivativo . . . . .	61
6.5	Controllo braccia . . . . .	63
6.5.1	Controllo proporzionale derivativo . . . . .	63
6.5.2	Controllo feed-forward con coppia pre-computata . . . . .	67
6.5.3	Controllo in dinamica inversa . . . . .	68
	Test Kp e Kd . . . . .	71
6.5.4	Controllo robusto . . . . .	72
	Analisi manovellismo . . . . .	79
6.6	Confronto approcci controllori . . . . .	80
<b>7</b>	<b>Conclusioni</b>	<b>82</b>
7.1	Confronto modellazione teorica-pratica . . . . .	82
7.2	Sviluppi futuri . . . . .	82



---

# 1 Introduzione

## 1.1 Obiettivo

L’obiettivo di questa tesi riguarda lo studio teorico e l’implementazione pratica di approcci di controllo centralizzato ad un manipolatore a cinematica parallela (5 gradi di libertà) con l’obiettivo di andare a trovare il migliore. In primis verrà eseguita una modellazione fisica e teorica del manipolatore, successivamente mediante un’attività sperimentale che comprende l’implementazione degli schemi di controllo analizzati e la generazione di traiettorie bidimensionali e tridimensionali, si andranno ad analizzare i risultati ottenuti.

## 1.2 Stato dell’arte

Il lavoro di tesi è stato svolto in collaborazione con il laboratorio di meccatronica dell’università degli studi di Bergamo. Alla base di questa tesi vi è il manipolatore PKM (*parallel kinematic manipulator*) costruito nel 2013. Un manipolatore parallelo è un sistema meccanico che utilizza ”catene” seriali per supportare un *end-effector*, ogni catena solitamente è corta, semplice e di conseguenza può essere rigida rispetto a movimenti non voluti rispetto ad un manipolatore seriali. La movimentazione e la flessibilità di un *joint* è vincolata dall’effetto delle altre catene, questo rende il manipolatore rigido rispetto alle sue componenti.

In particolare, nel nostro caso il manipolatore è composto da quattro link, due distali ovvero motorizzati e due prossimali, all’estermità è connessa una vite che consente traslazione e rotazione nell’asse Z, in questa configurazione il robot arriva ad avere 5 gradi di libertà. Escludendo l’introduzione, il lavoro di tesi sarà articolato in sette capitoli: nel secondo capitolo verrà presentata l’analisi cinematica del manipolatore, in particolare verranno analizzate sia cinematica diretta che inversa, nel terzo capitolo si parlerà di dinamica, punti di singolarità e manipolabilità del robot; il quarto capitolo mostrerà la modellazione dell’*end-effector*, nel nostro caso la vite; il quinto capitolo andrà a presentare tutte le tecnologie implementate a livello pratico, e per concludere nel sesto capitolo verrà presentato il sistema reale,

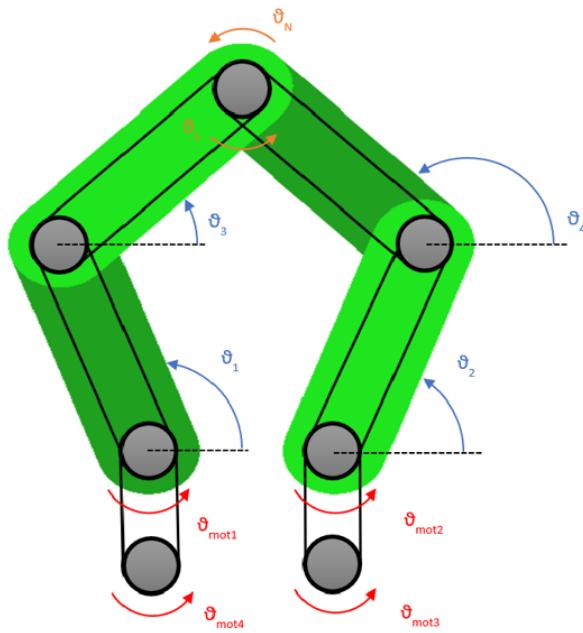


Figura 1.1: Robot PKM

includendo la struttura, le modalità di comunicazione, il software implementato, il sistema di controllo, l’interfaccia grafica ed i problemi riscontrati con le relative soluzioni. Infine, nel settimo capitolo, verranno esposte le conclusioni, grazie ad un confronto ottenuto tra risultati teorici e quelli pratici.

### 1.3 Ambiti applicativi

Grazie alle loro caratteristiche, i manipolatori a cinematica parallela vengono utilizzati in diversi ambiti, di seguito è riportato solo qualche esempio:

- Simulatori di volo
- Simulatori di guida
- Allineamento e posizionamento della fibra ottica
- Ambito medicale
- Assemblamento *PCB*
- Operazioni di *pick & place*

## 1.4 Software modellazione teorica

Per la modellazione teorica del manipolatore, abbiamo bisogno di utilizzare strumenti software specifici; in particolare verranno utilizzati Matlab e Adams, il primo ci consentirà di realizzare un modello software del manipolatore, il secondo, sempre a partire da un modello ci servirà a validare i dati ottenuti dal primo in modo da verificare la loro correttezza.

### 1.4.1 Matlab

Matlab, abbreviazione di *Matrix Laboratory*, è una piattaforma di calcolo ottimizzata nella risoluzione di problemi tecnici. Matlab è un linguaggio ad alte prestazioni per la computazione tecnica, comprende: computazione, visualizzazione e programmazione in un ambiente di facile utilizzo dove i problemi e le soluzioni vengono espressi mediante una notazione matematica, gli utilizzi tipici riguardano: matematica, sviluppo di algoritmi, modellazione, simulazione, prototipazione, analisi dei dati, intelligenza artificiale, verifiche di computazione. La base di matlab è un vettore che non ha bisogno di dimensioni, in questo modo permette la risoluzione di molti problemi di computazione, specialmente quelli in formulazione vettoriale e matriciale velocemente, senza dover ricorrere all'utilizzo di linguaggi come il C. Matlab inoltre possiede delle *toolbox* ovvero moduli aggiuntivi che permettono di specializzarsi in un campo, in particolare sono insieme di funzioni MATLAB che estendono l'ambiente, permettendogli di risolvere particolari classi di problemi, esempi di moduli sono reti neurali, processamento di segnali, sistemi di controllo.

Nel nostro caso abbiamo definito il modello teorico del robot, abbiamo poi calcolato cinematica, dinamica e punti di singolarità, nei capitoli successivi verranno mostrate le operazioni fatte.

### 1.4.2 Adams

Adams è un software utilizzato nel campo della dinamica *multibody*, in particolare nell'analisi dei modelli, infatti dopo che è stato progettato un modello può essere importato in adams ed è possibile fare analisi, simulazioni e validazioni, andando

quindi a simulare la fisica del mondo reale. Adams è anche ottimizzato per problemi di grandi dimensioni.

Il software ha una GUI completa, infatti consente anche di disegnare direttamente il modello nello spazio tridimensionale o di importare file come STEP e IGS. I *joint* possono essere aggiunti tra due corpi per vincolare il loro movimento, inoltre al sistema possono essere passati input come velocità, forze e condizioni iniziali. Adams simula il comportamento del sistema al variare del tempo, consente anche l'animazione e la computazione di proprietà come le forze, le inerzie e le accelerazioni, è anche possibile nel sistema includere elementi complessi dinamicamente come per esempio molle, corpo flessibili, contatto tra corpi. È inoltre possibile esportare tutti i dati in formato tabellare per fare analisi successive.

Per quanto riguarda il nostro caso, abbiamo utilizzato Adams per modellare il robot, assegnargli le coppie e confrontare i valori della cinematica e dinamica con quelli ottenuti da Matlab, nei capitoli successivi verrà presentato un confronto tra questi dati.

## 2 Cinematica Manipolatore

L'obiettivo di questa sezione è quello di andare ad illustrare le metodologie che ci hanno consentito di ottenere sia la cinematica diretta che quella inversa, entrambe per posizione, velocità ed accelerazione.

Prima di proseguire nei paragrafi seguenti andiamo a definire una tabella con i principali parametri del robot: Tutti e quattro i link hanno lunghezza, massa,

Nome	Descrizione	Valore
$l[m]$	lunghezza link	0.25
$m[kg]$	massa link	2.9
$c_m[m]$	posizione centro di massa	1.25
$J_r[kg \cdot m^2]$	momento d'inerzia baricentrico	$5.22 \cdot 10^{-2}$
$d[m]$	lunghezza semitelaio	0.09

Tabella 1: Parametri manipolatore

posizione del centro di massa e momento di inerzia uguali, per questo si è deciso di rappresentare i dati una sola volta.

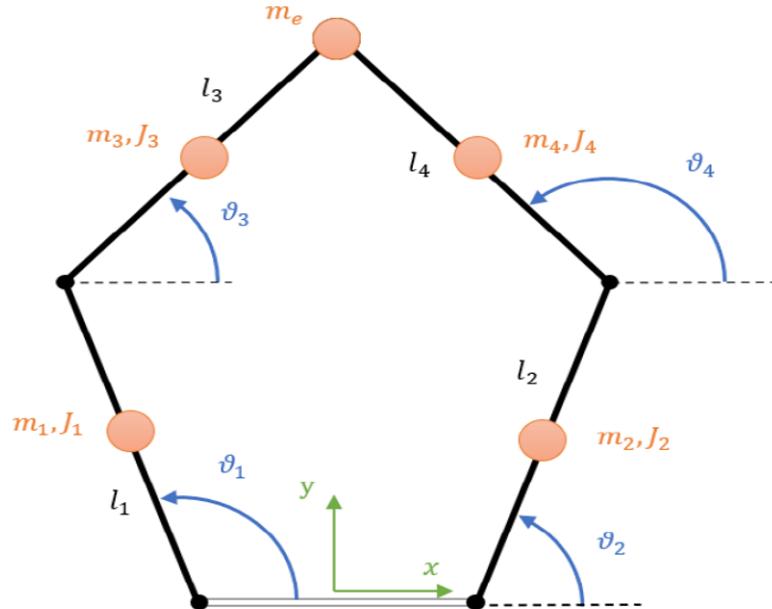


Figura 2.1: Rappresentazione fisica del robot

## 2.1 Cinematica Diretta

La cinematica diretta si occupa di trovare il legame tra i parametri interni del robot e la posa che esso assume, per posa si intende la posizione e l'orientamento. In questa sezione andremo ad analizzare la cinematica diretta di posizione, velocità ed accelerazione.

### 2.1.1 Posizione

Nella cinematica diretta di posizione, a partire dal robot e da  $\theta_1$  e  $\theta_2$ , riusciamo a ricavare la posizione dei link non motorizzati, i loro angoli, che sono rispettivamente  $\theta_3$  e  $\theta_4$  e la posizione  $[x, y]$  dell'*end-effector*. L'approccio utilizzato per il calcolo della cinematica diretta è stato quello delle equazioni alle circonferenze, in particolare vengono definite due equazioni

- Circonferenza centrata in E1 che passa per l'*end-effector* e la base del primo link
- Ciconferenza centrata in E2 che passa per l'*end-effector* e la base del secondo link

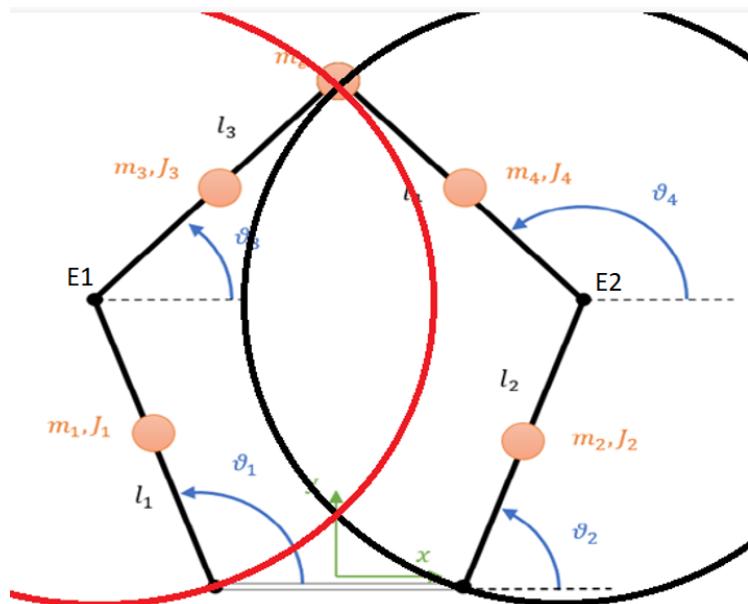


Figura 2.2: Equazioni alle circonferenze

Dalla combinazione di queste due equazioni otteniamo il seguente sistema:

$$\begin{cases} (x - \frac{l}{2} - l \cos \theta_1)^2 + (y - l \sin \theta_1)^2 = l^2 \\ (x + \frac{l}{2} - l \cos \theta_2)^2 + (y - l \sin \theta_2)^2 = l^2 \end{cases} \quad (2.1)$$

Da queste, andando a sviluppare i calcoli possiamo definire i parametri:

$$A = l^2(\sin \theta_2 - \sin \theta_1)^2 + (-2d - l(\cos \theta_2 - \cos \theta_1))^2$$

$$B = -2l^2d(\sin \theta_2 - \sin \theta_1)(\cos(\theta_2 + \theta_1) + l(\sin \theta_2 - \sin \theta_1)(-2d - l(\cos \theta_2 - \cos \theta_1))) \\ (-2d - 2l \cos \theta_2) - 2l \sin \theta_2(-2d - l(\cos \theta_2 - \cos \theta_1))^2$$

$$C = l^2d^2(\cos \theta_2 + \cos \theta_1)^2 - ld(\cos \theta_2 + \cos \theta_1)(-2d - l(\cos \theta_2 - \cos \theta_1)) \\ (-2d - 2l \cos \theta_2) + (d^2 + 2dl \cos \theta_2)(-2dl(\cos \theta_2 - \cos \theta_1))^2$$

Il passo successivo è quello di ricavare la posizione  $P = [x, y]$  dell'end-effector; si può procedere grazie alla formula risolutiva delle equazioni di secondo grado

$$s_{1,2}{}^1 = \frac{-b \pm \Delta}{2a}$$

Si può notare facilmente che la formula fornisce due risultati diversi, in uno caso quando si utilizza  $+\Delta$  e l'altro quando si usa  $-\Delta$ , è normale perché secondo il teorema fondamentale dell'algebra possono esserci fino a due soluzioni, in questo caso una coincidente con la posizione dell'end-effector e l'altra proiettata sulla base del semi-telaio.

$$x = \frac{y \cdot l(\sin \theta_2 - \sin \theta_1) - l \cdot d(\cos \theta_2 + \cos \theta_1)}{-2d - l(\cos \theta_2 - \cos \theta_1)}, y = \frac{-B + \sqrt{B^2 - 4AC}}{2A} \quad (2.2)$$

Possiamo poi andare a trovare le posizioni dei link distali mediante relazioni geometriche nel seguente modo:

$$E_{1X} = -d + l \cdot \cos \theta_1 \quad E_{1Y} = l \cdot \sin \theta_1$$

---

<sup>1</sup> $\Delta = \sqrt{b^2 - 4ac}$

## 2.1 Cinematica Diretta

---

e

$$E_{2X} = d + l \cdot \cos \theta_2 \quad E_{2Y} = l \cdot \sin \theta_2$$

Adesso che abbiamo  $E_1$  ed  $E_2$  possiamo andare a trovare gli angoli  $\theta_3, \theta_4$  in funzione di  $\theta_1$  e  $\theta_2$

$$\theta_3 =$$

$$2 \cdot \operatorname{tg}^{-1} \frac{\sqrt{(\sin \theta_2 - \sin \theta_1) + \frac{1}{2}((\cos \theta_2 - \cos \theta_1 + \frac{18}{25})^2 + (\frac{(\sin \theta_1 - \sin \theta_2)^2}{2})^2 + (\sin \theta_1 - \sin \theta_2)^2)}}{(\cos \theta_2 - \cos \theta_1) + \frac{(\cos \theta_2 - \cos \theta_1 + \frac{18}{25})^2}{2} + \frac{(\sin \theta_1 - \sin \theta_2)^2}{2} + \frac{18}{25}}$$

$$\theta_4 =$$

$$-2 \cdot \operatorname{tg}^{-1} \frac{\sqrt{(\sin \theta_2 - \sin \theta_1) + (\frac{1}{2}(\cos \theta_2 - \cos \theta_1 + \frac{18}{25})^2 + (\frac{(\sin \theta_1 - \sin \theta_2)^2}{2})^2 + (\sin \theta_1 - \sin \theta_2)^2)}}{(\cos \theta_1 - \cos \theta_2) + \frac{(\cos \theta_2 - \cos \theta_1 + \frac{18}{25})^2}{2} + \frac{(\sin \theta_1 - \sin \theta_2)^2}{2} + \frac{18}{25}}$$

Di conseguenza, alla fine della cinematica diretta siamo riusciti ad ottenere i parametri:

$$[x, y, E1, E2, \theta_3, \theta_4]$$

tutti espressi in funzione di  $\theta_1$  e  $\theta_2$ .

### 2.1.2 Velocità

Una volta ottenute le posizioni possiamo passare alle velocità, mediante la cinematica diretta di velocità possiamo ricavare le velocità sulle coordinate x e y dell'*end-effector*. Possiamo infine definire una jacobiana che ci permette di trovare il rapporto appena espresso.

Per semplicità di calcoli, andiamo a definire:

$$N_{21} = -l \sin \theta_1 (x + d - l \cdot \cos \theta_1 + l \cdot \cos \theta_1 (y - l \sin \theta_1))$$

$$N_{22} = -l \cos \theta_2 \cdot \frac{y - l \sin \theta_2 (x + d - l \cos \theta_1)}{x - d - l \cos \theta_2} + l \sin \theta_2 (x + d - l \cos \theta_1)$$

$$D_2 = \frac{y - l \sin \theta_2 (x + d - l \cos \theta_1)}{x - d - l \cos \theta_2}$$

Una volta definiti e calcolati questi valori possiamo andare a costruire i termini della jacobiana:

$$\begin{aligned} J_{11} &= -\frac{y - l \sin \theta_2}{x - d - l \cos \theta_2} \cdot J_{21} \\ J_{12} &= -\frac{y - l \sin \theta_2}{x - d - l \cos \theta_2} \cdot J_{22} - l \sin \theta_2 + \frac{y - l \sin \theta_2}{x - d - l \cos \theta_2} \cdot l \cos \theta_2 \\ J_{21} &= \frac{N_{21}}{D_2} \\ J_{22} &= \frac{N_{22}}{D_2} \end{aligned}$$

Posizionando i termini della matrice possiamo quindi definire  $J$  come:

$$J = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \quad (2.3)$$

Avendo quindi definito la jacobiana possiamo ricavare la velocità dell'*end-effector*  $\dot{P} = [\dot{x}, \dot{y}]$ , nel seguente modo:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = J \cdot \dot{\Theta} = \begin{bmatrix} J_{11} & J_{12} \\ J_{21} & J_{22} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} J_{11}\dot{\theta}_1 + J_{12}\dot{\theta}_2 \\ J_{21}\dot{\theta}_1 + J_{22}\dot{\theta}_2 \end{bmatrix} \quad (2.4)$$

### 2.1.3 Accelerazione

Anche per quanto riguarda l'accelerazione il processo simile a quanto visto nei paragrafi precedenti, in questo caso a partire da tutti i parametri precedentemente ricavati e da  $\ddot{\Theta}$  composto da  $\ddot{\theta}_1$  e  $\ddot{\theta}_2$  ricaviamo le accelerazioni all'*end-effector*. L'idea base è quella di risolvere la seguente equazione:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \end{bmatrix} = J \ddot{\Theta} + J \ddot{\Theta} = J \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + J \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} \quad (2.5)$$

Andando a svolgere le derivate ed i calcoli prima di poter trovare le accelerazioni definiamo:

$$\begin{aligned} A_{acc} = & \dot{x}^2 + 2l \sin \theta_2 \dot{x} \dot{\theta}_2 + (l \sin \theta_2 \dot{\theta}_2)^2 + (x - d - l \cos \theta_2)(l \cos \theta_2 \ddot{\theta}_2 + l \sin \theta_2 \ddot{\theta}_2) + \\ & \dot{y}^2 - 2l \cos \theta_2 \dot{y} \dot{\theta}_2 + (l \cos \theta_2 \dot{\theta}_2)^2 + (y - l \sin \theta_2)(l \sin \theta_2 \dot{\theta}_2^2 - l \cos \theta_2 \dot{\theta}_2^2) \end{aligned}$$

$$B_{acc} = \dot{x}^2 + 2l \sin \theta_1 \dot{x} \dot{\theta}_1 + (l \sin \theta_1 \dot{\theta}_1)^2 + (x + dl \cos \theta_1)(l \cos \theta_1 \dot{\theta}_1^2 + l \sin \theta_1 \ddot{\theta}_1) + \\ \dot{y}^2 - 2l \cos \theta_1 \dot{y} \dot{\theta}_1 + (l \cos \theta_1 \dot{\theta}_1)^2 + (y - l \sin \theta_1)(l \sin \theta_1 \dot{\theta}_1^2 - l \cos \theta_1 \ddot{\theta}_1)$$

Infine, andiamo a trovare  $\ddot{P} = [\ddot{x}, \ddot{y}]$ , nel seguente modo:

$$\ddot{x} = -\frac{\ddot{y}(y - l \sin \theta_1)}{x + d - l \cos \theta_1} \quad (2.6)$$

$$\ddot{y} = \frac{\frac{B_{acc} \cdot (x - d - l \cos \theta_2)}{x + d l \cos \theta_1 - A_{acc}}}{y - l \sin \theta_2 - \frac{x - d - l \cos \theta_2}{(x + d - l \cos \theta_1) \cdot (y - l \sin \theta_1)}} \quad (2.7)$$

Alla fine della cinematica diretta, a partire dai vettori delle posizioni, velocità ed accelerazioni ( $\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2, \ddot{\theta}_1, \ddot{\theta}_2$ ) siamo riusciti ad ottenere le posizioni, velocità ed accelerazioni riferite all'*end-effector*.

## 2.2 Cinematica inversa

Il problema della cinematica inversa consiste nel ricavare i valori degli angoli da assegnare ai parametri del robot per riuscire a seguire una determinata legge di moto o traiettoria a partire dalla posizione alle estremità, in questo caso l'*end-effector*. Anche l'analisi della cinematica inversa è stata fatta per posizione, velocità ed accelerazione.

### 2.2.1 Posizione

Nella cinematica inversa di posizione, a partire dalla posizione dell'*end-effector*  $P = [x, y]$  andiamo a ricavare  $\theta_1$  e  $\theta_2$ . Definiamo i seguenti parametri:

$$p = 2dl + 2xl$$

$$e = 2yl$$

$$f = x^2 + d^2 + y^2 + 2px$$

che serviranno per il calcolo di  $\theta_1$ , e:

$$a = -2dl + 2xl$$

$$b = 2yl$$

$$c = x^2 + d^2 + y^2 - 2xd$$

che serviranno per il calcolo di  $\theta_2$ .

Procediamo quindi con i calcoli, andando a trovare le soluzioni:

$$\theta_1 = 2 \arctan \frac{e + \sqrt{p^2 + e^2 - f^2}}{p + f} \quad (2.8)$$

e

$$\theta_2 = 2 \arctan \frac{b - \sqrt{a^2 + b^2 - c^2}}{a + c} \quad (2.9)$$

Si può notare che sia per  $\theta_1$  che per  $\theta_2$  la somma di termini sotto la radice quadrata può dare un risultato reale o un risultato complesso, in caso che esca un risultato reale non c'è alcun problema, ma nel caso in cui  $p^2 + e^2 - f^2 \leq 0$  oppure  $a^2 + b^2 - c^2 \leq 0$  potrebbe verificarsi un caso di singolarità<sup>2</sup>.

### 2.2.2 Velocità

Per quanto riguarda il calcolo della cinematica inversa in velocità abbiamo bisogno delle velocità  $\dot{P} = [\dot{x}, \dot{y}]^T$  e della Jacobiana che lega  $\theta_1$  e  $\theta_2$ . Prendiamo l'equazione 2.3 andiamo poi ad invertirla e ricaviamo le velocità  $\dot{\theta}_1$  e  $\dot{\theta}_2$  nel seguente modo:

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = J^{-1} \begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \quad (2.10)$$

Con

$$J^{-1} = \frac{1}{J_{11}J_{22} - J_{12}J_{21}} \begin{bmatrix} J_{22} & -J_{12} \\ J_{21} & J_{11} \end{bmatrix} \quad (2.11)$$

---

<sup>2</sup>I punti di singolarità sono punti nei quali il manipolatore non si comporta in modo standard, potrebbero causarsi anche rotture, verranno descritti in modo approfondito nel capitolo riguardante la dinamica

Siamo quindi riusciti ad ottenere le velocità degli angoli a partire dalle velocità all'end-effector.

### 2.2.3 Accelerazione

Anche per le accelerazioni la logica di funzionamento è la medesima, volendo trovare  $\ddot{\theta}_1$  e  $\ddot{\theta}_2$  a partire da  $\ddot{P} = [\ddot{x}, \ddot{y}]^T$  implica che dobbiamo trovare la soluzione a:

$$\begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} = J^{\ddot{-1}} \dot{P} + J^{-1} \ddot{P} = J^{-1} (\ddot{P} - J \dot{\Theta}) \quad (2.12)$$

---

### 3 Dinamica Manipolatore

Il modello dinamico del manipolatore ci fornisce una descrizione matematica della relazione che è instaurata tra le forze agenti sul robot (generalizzate) ed il movimento prodotto dalla sua struttura, cioè le configurazioni che assume nel tempo. Inizialmente, nel calcolo della dinamica sono stati usati tre metodi diversi, il metodo delle azioni vincolari, il metodo di Lagrange e quello dei lavori virtuali. Si è poi deciso di proseguire esclusivamente con il PLV. L'obiettivo di questa sezione è quello di mostrare l'approccio e i risultati ottenuti per il calcolo della dinamica diretta ed inversa.

#### 3.1 Prerequisiti per il calcolo della dinamica

Prima di andare ad analizzare i metodi utilizzati, è importante andare a ricavare tutte le matrici delle quali avremo bisogno, in particolare è necessario andare a definire delle matrici che ci permettano di ottenere  $\theta_3$  e  $\theta_4$  in funzione di  $\theta_1$  e  $\theta_2$ .

##### 3.1.1 Jacobiana $J_{34}$ e calcolo $\dot{\theta}_3, \dot{\theta}_4$

A partire da  $\theta_1, \theta_2$  possiamo, mediante la cinematica diretta ottenere  $\theta_3$  e  $\theta_4$ , unendo queste quattro componenti e considerando anche le velocità  $\dot{\theta}_1$  e  $\dot{\theta}_2$  possiamo andare a ricavare la matrice  $J_{34}$  che esprime  $\theta_3$  e  $\theta_4$  in funzione di  $\theta_1, \theta_2$  e possiamo ricavare anche  $\dot{\theta}_3, \dot{\theta}_4$ , per far questo andiamo a definire le seguenti quantità:

$$N_{13} = \frac{\cos \theta_4}{\sin \theta_4} \sin \theta_1 - \cos \theta_1$$

$$N_{23} = \cos \theta_2 - \frac{\cos \theta_4}{\sin \theta_4} \sin \theta_2$$

$$D_{13} = \cos \theta_3 - \frac{\cos \theta_4}{\sin \theta_4} \sin \theta_3$$

Da queste tre formule possiamo andare a ricavare  $\dot{\theta}_3$  nel seguente modo:

$$\dot{\theta}_3 = \frac{\dot{\theta}_1 N_{13}}{D_{13}} + \frac{\dot{\theta}_2 N_{23}}{D_{13}} \quad (3.1)$$

Proseguiamo ora definendo le equazioni che ci serviranno per calcolare  $\dot{\theta}_4$ :

$$\begin{aligned} N_{14} &= \frac{\sin \theta_1 \cos \theta_3}{\sin \theta_3} - \frac{\cos \theta_4}{\sin \theta_4} + \frac{\cos \theta_4}{\sin \theta_4} \sin \theta_1 - \cos \theta_1 \\ N_{24} &= -\frac{\sin \theta_2 \cos \theta_3}{\sin \theta_3} - \frac{\cos \theta_4}{\sin \theta_4} - \frac{\cos \theta_4}{\sin \theta_4} \sin \theta_2 + \cos \theta_2 \\ D_{14} &= \frac{\sin \theta_4 \cos \theta_3}{\sin \theta_3} - \frac{\cos \theta_4}{\sin \theta_4} \end{aligned}$$

Otteniamo quindi:

$$\dot{\theta}_4 = \dot{\theta}_1 \frac{N_{14}}{D_{14}} + \dot{\theta}_2 \frac{N_{24}}{D_{14}} \quad (3.2)$$

Il passo finale è quello di andare a rappresentare la matrice jacobiana che lega le velocità  $\dot{\theta}_3$  e  $\dot{\theta}_4$  con le velocità in ingresso al manipolatore:

$$J_{34} = \begin{bmatrix} \frac{N_{13}}{D_{13}} & \frac{N_{23}}{D_{13}} \\ \frac{N_{14}}{D_{14}} & \frac{N_{24}}{D_{14}} \end{bmatrix} \quad (3.3)$$

### 3.1.2 Jacobiana $J_{34}$ e calcolo di $\ddot{\theta}_3, \ddot{\theta}_4$

Per andar ad ottenere la jacobiana finale, ed i valori delle accelerazioni sui link distali occorre derivare tutti gli elementi visti in precedenza, in particolare:

$$\begin{aligned} \dot{N}_{13} &= \frac{-1}{\sin^2 \theta_4 \cdot \dot{\theta}_4 \sin \theta_1} + \frac{\cos \theta_4}{\sin \theta_4} \cos \theta_1 \dot{\theta}_1 + \sin \theta_1 \dot{\theta}_1 \\ \dot{N}_{23} &= \frac{1}{\sin^2 \theta_4 \cdot \dot{\theta}_4 \sin \theta_2} - \frac{\cos \theta_4}{\sin \theta_4} \cos \theta_2 \dot{\theta}_2 - \sin \theta_2 \dot{\theta}_2 \\ \dot{D}_{13} &= -\sin \theta_3 \dot{\theta}_3 + \frac{1}{\sin^2 \theta_4} \dot{\theta}_4 \sin \theta_3 - \frac{\cos \theta_4}{\sin \theta_4} \cos \theta_3 \dot{\theta}_3 \\ \dot{N}_{14} &= \cos \theta_1 \dot{\theta}_1 \left( \frac{\cos \theta_3}{\sin \theta_3} - \frac{\cos \theta_4}{\sin \theta_4} \right) + \sin \theta_1 \left( \frac{1}{\sin^2 \theta_3} \dot{\theta}_3 + \frac{1}{\sin^2 \theta_4} \dot{\theta}_4 \right) + \\ &\quad + \frac{-1}{\sin^2 \theta_4} \dot{\theta}_4 \sin \theta_1 + \cot \theta_4 \cos \theta_1 \dot{\theta}_1 + \sin \theta_1 \dot{\theta}_1 \\ \dot{N}_{24} &= -\cos \theta_2 \dot{\theta}_2 \left( \frac{\cos \theta_3}{\sin \theta_3} - \frac{\cos \theta_4}{\sin \theta_4} \right) - \sin \theta_2 \left( \frac{-1}{\sin^2 \theta_3} \dot{\theta}_3 + \frac{1}{\sin^2 \theta_4} \dot{\theta}_4 \right) - \\ &\quad - \frac{-1}{\sin^2 \theta_4} \dot{\theta}_4 \sin \theta_2 - \cot \theta_4 \cos \theta_2 \dot{\theta}_2 - \sin \theta_2 \dot{\theta}_2 \end{aligned}$$

$$\dot{D}_{14} = \cos \theta_4 \dot{\theta}_4 (\cot \theta_3 - \cot \theta_4) + \sin \theta_4 \left( \frac{-1}{\sin^2 \theta_3} \dot{\theta}_3 + \frac{1}{\sin^2 \theta_4} \dot{\theta}_4 \right)$$

Esprimendo la matrice jacobiana  $\dot{J}_{34}$  in funzione dei parametri appena trovati scriviamo:

$$\dot{J}_{34} = \begin{bmatrix} \frac{\dot{N}_{13}D_{13}-N_{13}\dot{D}_{13}}{D_{13}^2} & \frac{\dot{N}_{23}D_{13}-N_{23}\dot{D}_{13}}{D_{13}^2} \\ \frac{\dot{N}_{14}D_{14}-N_{14}\dot{D}_{14}}{D_{14}^2} & \frac{\dot{N}_{24}D_{14}-N_{24}\dot{D}_{14}}{D_{14}^2} \end{bmatrix} \quad (3.4)$$

Per concludere andiamo a trovare:

$$\begin{bmatrix} \ddot{\theta}_3 \\ \ddot{\theta}_4 \end{bmatrix} = J_{34} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} + J_{34} \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} \quad (3.5)$$

### 3.1.3 Matrici di inerzia

Per trovare la soluzione all'equazione del PLV introduciamo le matrici che hanno avuto un ruolo fondamentale nel calcolo:

$$\begin{aligned} J_1 &= \begin{bmatrix} -0.5l \sin \theta_1 & 0 \\ 0.5l \cos \theta_1 & 0 \end{bmatrix} \Rightarrow \dot{J}_1 = \begin{bmatrix} -0.5l \cos \theta_1 \dot{\theta}_1 & 0 \\ -0.5l \sin \theta_1 \dot{\theta}_1 & 0 \end{bmatrix} \\ J_2 &= \begin{bmatrix} 0 & -0.5 * l * \sin \theta_2 \\ 0 & 0.5 * l * \cos \theta_2 \end{bmatrix} \Rightarrow \dot{J}_2 = \begin{bmatrix} 0 & -0.5l \cos \theta_2 \dot{\theta}_2 \\ 0 & -0.5l \sin \theta_2 \dot{\theta}_2 \end{bmatrix} \\ J_3 &= \begin{bmatrix} -l \sin \theta_1 + 0.5 \sin \theta_3 \cdot J_{34}(1,1) & -0.5l \sin \theta_3 \cdot J_{34}(1,2) \\ l \cos \theta_1 + 0.5 \cos \theta_3 \cdot J_{34}(1,1) & 0.5l \cos \theta_3 \cdot J_{34}(1,2) \end{bmatrix} \\ J_4 &= \begin{bmatrix} -0.5l \sin \theta_4 \cdot J_{34}(2,1) & -l \sin \theta_2 + 0.5 \sin \theta_4 \cdot J_{34}(2,2) \\ 0.5l \cos \theta_4 \cdot J_{34}(2,1) & l \cos \theta_2 + 0.5 \cos \theta_4 \cdot J_{34}(2,2) \end{bmatrix} \\ J_E &= \begin{bmatrix} -l(\sin \theta_1 + \sin \theta_3 \cdot J_{34}(1,1)) & -l \sin \theta_3 \cdot J_{34}(1,2) \\ l(\cos \theta_1 + \cos \theta_3 \cdot J_{34}(1,1)) & l \cos \theta_3 \cdot J_{34}(1,2) \end{bmatrix} \end{aligned}$$

Importanti nel calcolo della dinamica saranno anche le derivate delle matrici che abbiamo appena visto, ovvero  $\dot{J}_3$ ,  $\dot{J}_4$ ,  $\dot{J}_E$ .

### 3.2 Principio dei lavori virtuali

Il lavoro virtuale è il lavoro svolto da una forza reale che agisce attraverso uno spostamento virtuale o da una forza virtuale che agisce attraverso uno spostamento reale. Uno spostamento virtuale è uno spostamento coerente con i vincoli della struttura, cioè che soddisfano le condizioni al contorno in corrispondenza degli appoggi. Una forza virtuale è un qualsiasi sistema di forze in equilibrio. Per problemi nei quali i corpi sono composti da membri interconnessi che si possono muovere relativamente gli uni rispetto agli altri, originando diverse configurazioni di equilibrio un buon metodo di analisi è quello del "principio dei lavori virtuali" conosciuto anche come PLV ci permette di ottenere una relazione relativamente semplice, è basato sul concetto di Lavoro sviluppato da una forza, ed inoltre ci consente di analizzare la stabilità di un sistema in equilibrio.

$$\sum_{i=1}^m F_j \delta q_j \quad (3.6)$$

### 3.3 Dinamica inversa

Il problema della dinamica inversa consiste nel determinare le coppie ai giunti necessarie per generare il movimento a partire da posizione, velocità ed accelerazione. Andando a sviluppare l'equazione dei principi virtuali troviamo le coppie dei link motorizzati nel seguente modo:

$$\begin{aligned} \delta\theta^T C = & \delta\theta^T I_2 \ddot{\theta} + \delta\theta^T J_{34}^T I_2 (J_{34} \ddot{\theta} + J_{34} \dot{\theta}) + \delta\theta^T \frac{25}{4} ml^2 \\ & \left( \begin{bmatrix} -\cos \theta_1 & -\sin \theta_1 \\ -\cos \theta_2 & -\sin \theta_2 \end{bmatrix} \dot{\theta}^2 + \begin{bmatrix} -\sin \theta_1 & \cos \theta_1 \\ -\sin \theta_2 & \cos \theta_2 \end{bmatrix} \ddot{\theta} \right) + \delta\theta^T J_{34}^T \frac{9}{4} l^2 (m + m_v) \\ & \left( \begin{bmatrix} -\cos \theta_3 & -\sin \theta_3 \\ -\cos \theta_4 & -\sin \theta_4 \end{bmatrix} J_{34} J_{34}^T \dot{\theta}^2 + \begin{bmatrix} -\sin \theta_3 & \cos \theta_3 \\ -\sin \theta_4 & \cos \theta_4 \end{bmatrix} (J_{34} \dot{\theta} + J_{34} \ddot{\theta}) \right) \end{aligned}$$

Semplificando e raccogliendo possiamo esprimere l'equazione come:

$$\tau = M \ddot{\theta} + K \dot{\theta} \quad (3.7)$$

Dove:

$$M = J_r I_2 + m(J_1^T J_1 + J_2^T J_2 + J_3^T J_3 + J_4^T J_4) + J_r J_{34}^T J_{34} + m_v J_E^T J_E \quad (3.8)$$

$$K = m(J_1^T \dot{J}_1 + J_2^T \dot{J}_2 + J_3^T \dot{J}_3 + J_4^T \dot{J}_4) + J_r J_{34}^T \dot{J}_{34} + m_v J_E^T \dot{J}_E \quad (3.9)$$

Sostituendo, possiamo andare ad esprimere l'equazione 3.7 nel seguente modo:

$$\begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix} = M \begin{bmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{bmatrix} + K \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \quad (3.10)$$

Per tutti i calcoli svolti fino ad adesso le coppie devono considerarsi prese all'end-effector

### 3.4 Dinamica diretta

Il problema della dinamica diretta invece consiste nel determinare le accelerazioni ai giunti a partire dalle coppie, dalla posizione e velocità iniziali di entrambi i link. Identifichiamo quindi  $\Theta$  come vettore delle condizioni iniziali, in particolare possiamo definirlo come segue:

$$\Theta = \begin{bmatrix} \theta_1(t_0) \\ \theta_2(t_0) \\ \dot{\theta}_1(t_0) \\ \dot{\theta}_2(t_0) \end{bmatrix}$$

Possiamo andare a calcolare  $\theta_3$  e  $\theta_4$  come abbiamo visto in precedenza nella sezione 2.1.1, e di conseguenza anche tutte le matrici viste nella sezione 3.1. Con tutti questi dati possiamo andare a ricalcolare le equazioni 3.8 e 3.9.

Andiamo ora a definire l'equazione della dinamica diretta andando ad invertire l'equazione 3.7 in questo modo:

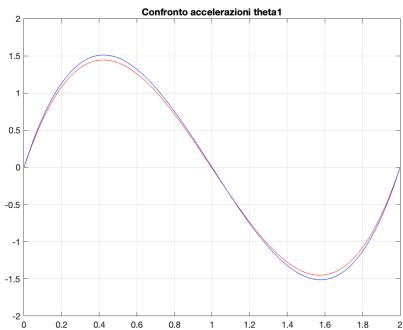
$$\ddot{\theta} = M^{-1}(-K\dot{\theta} + \tau) \quad (3.11)$$

Infine, da questa possiamo andare anche a calcolare velocità e posizioni integrando l'equazione.

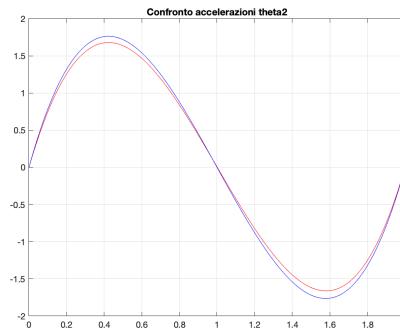
### 3.4 Dinamica diretta

---

Avendo sia il grafico della legge di moto iniziale relativa a posizione, velocità ed accelerazione assegnata agli angoli  $\theta_1$  e  $\theta_2$ , che le coppie, possiamo andare a calcolare la dinamica con le coppie e le condizioni iniziali e *plottare* il confronto tra queste due curve. In particolare andiamo a vedere il confronto su accelerazioni, velocità e posizioni sia per  $\theta_1$  che per  $\theta_2$ : Una volta calcolati tutti i parametri di nostro

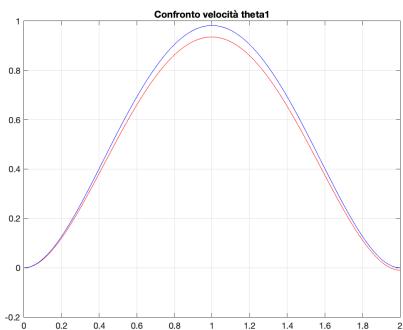


(a) Accelerazione  $\theta_1$

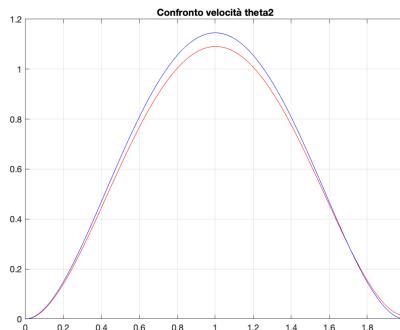


(b) Accelerazione  $\theta_2$

Figura 3.1: Confronto dinamica leggi di moto su accelerazioni

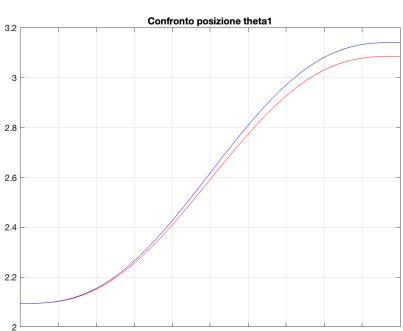


(a) Velocità  $\theta_1$

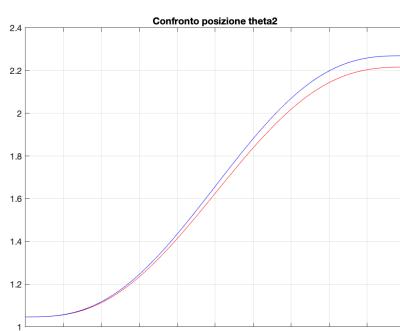


(b) Velocità  $\theta_2$

Figura 3.2: Confronto dinamica leggi di moto su velocità



(a) Posizione  $\theta_1$



(b) Posizione  $\theta_2$

Figura 3.3: Confronto dinamica leggi di moto su posizioni

interesse andiamo a costruire il modello simulink che sarà utile per una visione d'insieme:

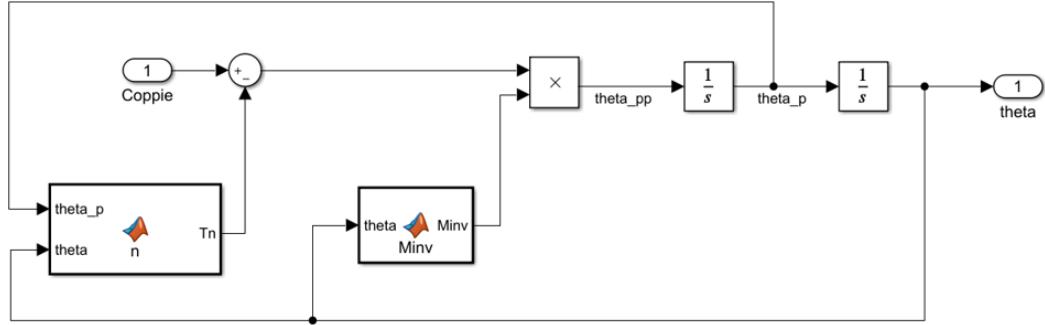


Figura 3.4: Modello simulink manipolatore

### Modellazione su Adams

Per quanto riguarda la modellazione su adams, è stato realizzato un prototipo del manipolatore composto da aste rigide, i due link motorizzati sono fissati mediante delle cerniere ed abbiamo anche la presenza dell'end-effector. Una volta definiti i

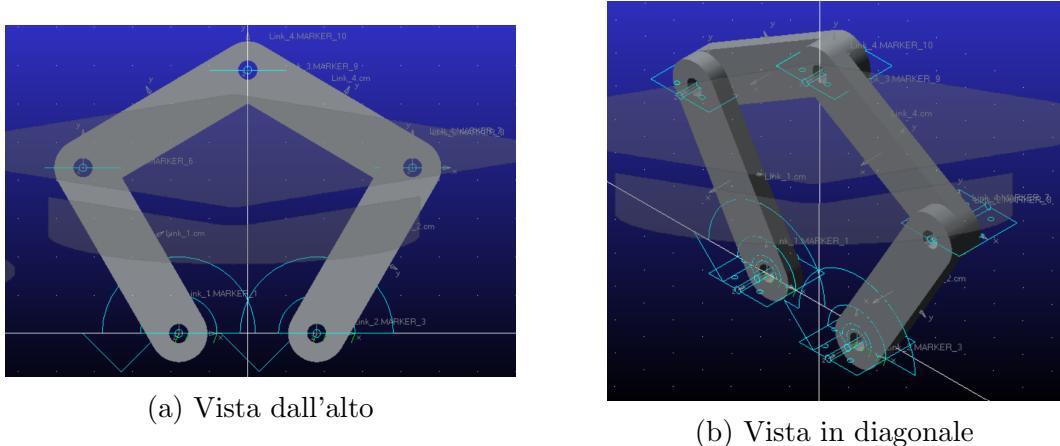


Figura 3.5: Modello Adams manipolatore 5R

vincoli e le modalità di movimento del manipolatore si è passati alla fase successiva ovvero quella dell'analisi del modello e della simulazione, andando ad assegnare leggi di moto al modello adams e verificare il suo comportamento rispetto al modello creato su simulink.

### Validazione e confronto

Le leggi di moto assegnate al modello adams sono state anche assegnate nella stessa maniera al modello simulink, in particolare sono state utilizzate: Una volta asse-

Motore	Legge di moto
Motore 1 ( $\theta_1$ )	Sinusoidale $\sin$
Motore 2 ( $\theta_2$ )	Sinusoidale $2 \sin$

Tabella 2: Leggi di moto validazione

gnata la legge ed eseguita la simulazione si è fatto un confronto: Andando poi a

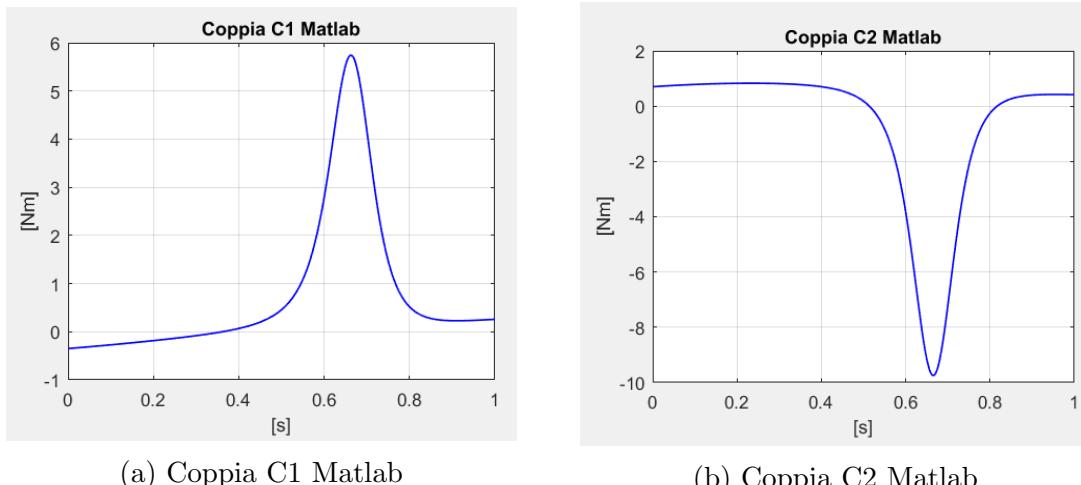


Figura 3.6: Coppie in uscita Simulink

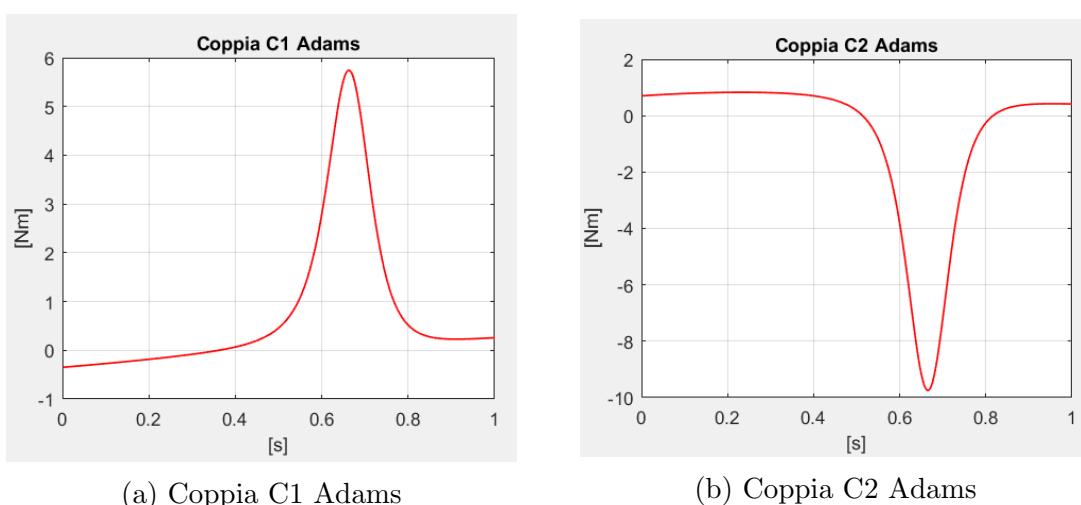


Figura 3.7: Coppie in uscita Adams

fare una differenza tra questi due grafici riusciamo a trovare l'andamento dell'err-

rore sulle coppie: Analizzando il grafico vediamo che l'errore è nell'ordine di  $10^{-7}$

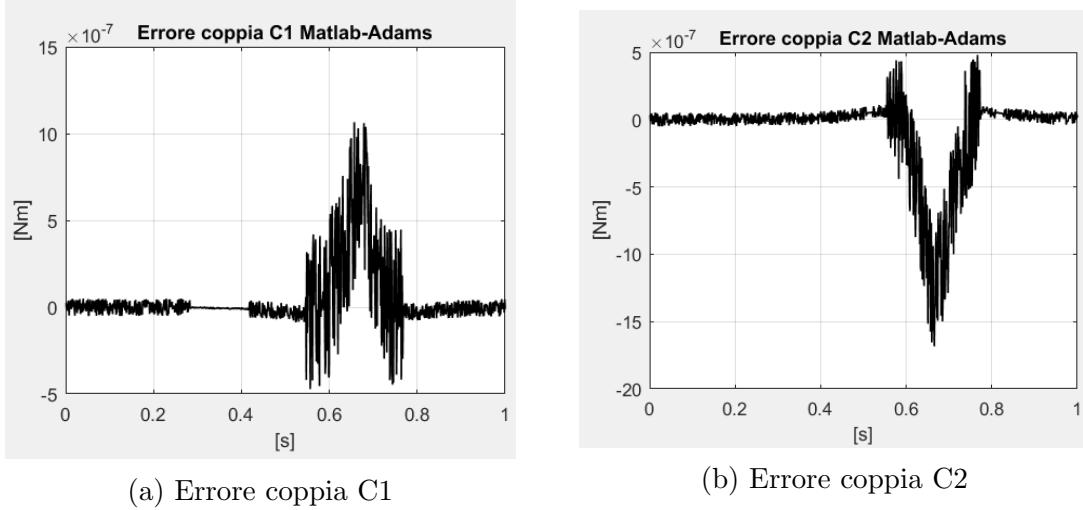


Figura 3.8: Errore Simulink-Adams

di conseguenza è possibile vedere che la validazione ha portato un risultato positivo in quanto le coppie del modello Simulink e le coppie del modello Adams sono praticamente identiche a parte un fattore d'errore dato dalle diverse modalità di calcolo/integrazione.

### 3.5 Punti di singolarità

Nell'ambito matematico, una singolarità è un punto nel quale un oggetto non è definito, oppure un punto nel quale l'oggetto non ha un comportamento normale, nel nostro caso i punti di singolarità saranno punti che andranno a delimitare lo spazio di lavoro del robot. Definiamo spazio di lavoro del robot tutto un insieme di punti nei quali il robot ha un funzionamento normale e non presenta problematiche.<sup>3</sup> Andando a considerare la foto vista nell'introduzione, possiamo trovare sei casi di punti di singolarità, in particolare però non sono punti ma sono traiettorie. Di conseguenza il robot avrà come spazio di lavoro, tutto lo spazio che è interno (delimitato) da queste traiettorie.

<sup>3</sup>Passando per un punto di singolarità il robot potrebbe aver problemi che potrebbero causare anche la rottura di parti meccaniche

**Primo e secondo caso**

In questo primo caso abbiamo  $\overrightarrow{CD}$  che è parallelo a  $\overrightarrow{DE}$ , schematicamente possiamo andarlo a rappresentare nel seguente modo

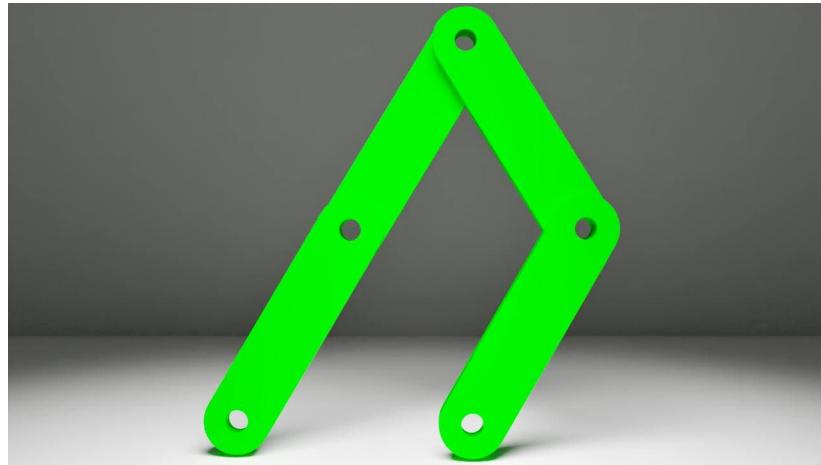


Figura 3.9: Caso 1 singolarità

Lasciando la x libera possiamo ricavare la y come:

$$y_1 = \sqrt{4l^2 - (x - d)^2} \quad (3.12)$$

Per quanto riguarda il secondo caso è molto simile al primo, la differenza sta nel fatto che abbiamo la catena  $\overrightarrow{AB}$  parallela a  $\overrightarrow{BC}$ . Il procedimento è simile a prima,

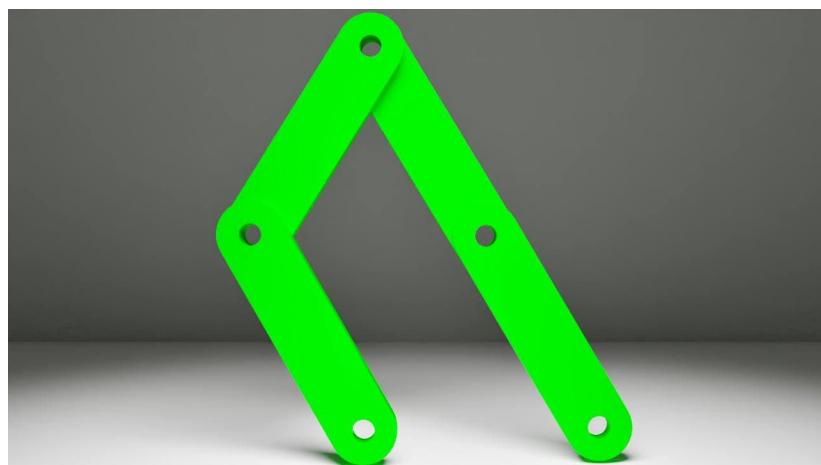


Figura 3.10: Caso 2 singolarità

lasciando sempre la x libera possiamo trovare la y come:

$$y_2 = \sqrt{4l^2 - (x + d)^2} \quad (3.13)$$

Entrambi i casi producono come risultato una circonferenza.

### Terzo caso

Il terzo caso di singolarità avviene quando i due link non motorizzati sono paralleli, questa configurazione aggiunge un grado di libertà al sistema, in quanto l'end-effector per muoversi ha necessità di una maggior coppia per riuscire a superare la situazione di stallo



Figura 3.11: Caso 3 singolarità

Per quanto riguarda la soluzione, andiamo a considerare  $\theta_1$  che varia da  $0$  a  $360^\circ$  e andiamo a cercare le coppie di valori  $[x, y]$  relative alla singolarità. Uscirà un'equazione di secondo grado, con i seguenti coefficienti:

$$a = l^2 \sin^2 \theta_1 + 4d^2 - 4dl \cos \theta_1 + l^2 \cos^2 \theta_1$$

$$b = 2l^3 \sin \theta_1 + 2dl^2 \sin \theta_1 \cos \theta_1 - 2dl \sin \theta_1 (2d - 2l \cos \theta_1)$$

$$c = l^2(l^2 + d^2 \cos^2 \theta_1 + 2ld \cos \theta_1) - 2dl(l + d \cos \theta_1)(2d - l \cos \theta_1) + d^2 - l^2(2d - l \cos \theta_1)^2$$

Risolviamo l'equazione di secondo grado:

$$\Delta = b^2 - 4ac$$

Andiamo a trovare le radici  $y$  di quest'equazione, definendo poi:  $sx = -b + l \cos \theta_1$  ed  $sy = l \sin \theta_1$  possiamo andare a trovare le soluzioni dell'equazione come:

$$x_3 = \left| \frac{x_{3p} + sx}{2} \right|, y_3 = \left| \frac{y_{3p} + sy}{2} \right| \quad (3.14)$$

Con

$$x_{3p} = \frac{l^2 + y_{3p}l \sin \theta_1 + ld \cos \theta_1}{2d - l \cos \theta_1}$$

#### Quarto e quinto caso

Il quarto e quinto caso sono casi di singolarità non realizzabili nella pratica, ma sono di interesse teorico. Il primo caso prevede che la posizione dell'end-effector coincida con la posizione del primo giunto motorizzato e nell'altro caso coinciderà con il secondo giunto motorizzato. Come soluzioni avremo semplicemente due punti e possiamo andare a calcolarli nei seguenti modi: Le soluzioni le possiamo trovare

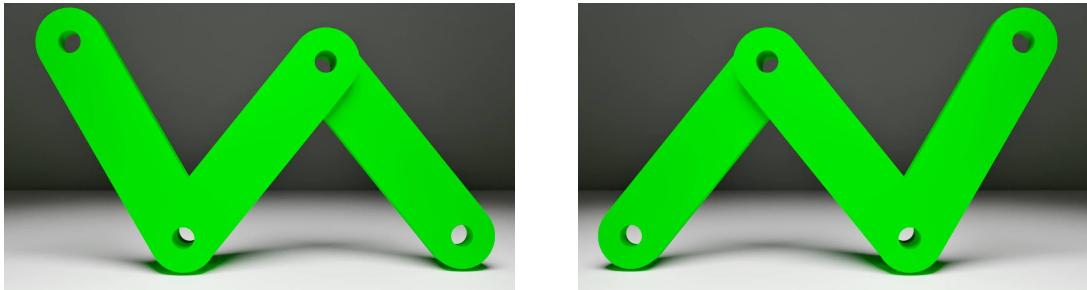


Figura 3.12: Caso 4 e 5 singolarità

impostando che la lunghezza della  $x$  vale nel quarto caso  $d$  e nel quinto  $-d$ , andando quindi a trovare le soluzioni come:

$$y_4 = \sqrt{-(x - d)^2} \quad (3.15)$$

e

$$y_5 = \sqrt{-(x + d)^2} \quad (3.16)$$

Andando ad unire tutti casi visti fino ad ora possiamo vederli visualmente nell'asse  $[x, y]$

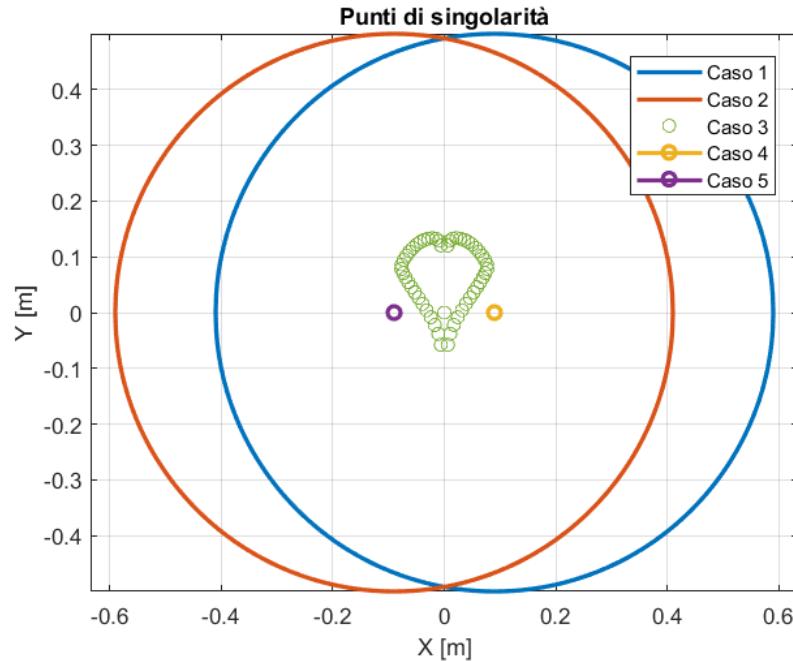


Figura 3.13: Punti di singolarità

## 3.6 Manipolabilità

La manipolabilità ci permette di avere una rappresentazione geometrica delle capacità che ha un punto del nostro sistema. Per andare a calcolarla abbiamo bisogno dell'equazione 2.3, vista nella sezione 2.1.2.

Andiamo a definire la matrice

$$J_{man} = J J^T \quad (3.17)$$

Da questa possiamo ricavare gli autovalori  $\Lambda$ . Definiamo poi l'indice di manipolabilità  $r$  come:

$$r = \frac{\max \lambda}{\min \lambda} \quad (3.18)$$

Questo numero può variare tra 1 e  $+\infty$ , più è piccolo e meno si rischia di andare in singolarità. Possiamo andare a rappresentare il numero di condizionamento tramite i grafici seguenti:

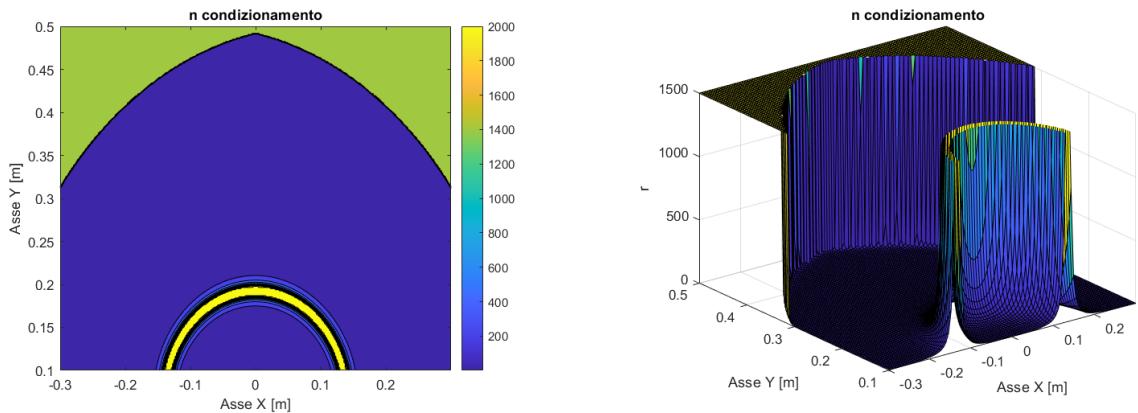


Figura 3.14: Numero di condizionamento

Nel primo grafico viene mostrato il piano x,y ed il numero di condizionamento è definito come una colormap, i punti di color blu sono quelli con un  $r$  piccolo ed in questi non siamo in condizioni di singolarità, invece quelli tendenti al verde/giallo sono i casi di singolarità che abbiamo visto prima. Nella seconda figura possiamo vedere la stessa rappresentazione però in tre dimensioni, utilizziamo l'asse z per rappresentare il numero di condizionamento; anche qua le zone di singolarità sono chiaramente visibili. In questo secondo grafico invece andiamo a concentrarci sulla zona di movimentazione ideale del manipolatore, più ci si avvicina allo zero, più il valore di  $r$  aumenta, questo per il fatto che stiamo raggiungendo una zona critica.

### 3.7 Workspace

Dalle analisi appena effettuate sui punti di singolarità e sul numero di condizionamento è stato possibile descrivere lo spazio di lavoro del manipolatore. In prima battuta ci si è concentrato sull'analizzare gli angoli di movimentazione dei giunti e i loro vincoli, in particolare il link sinistro con angolo  $\theta_1$  avrà una movimentazione di  $210^\circ$  al massimo, mentre il link destro con angolo  $\theta_2$  ha una movimentazione di  $180^\circ$ . Questa limitata mobilità è data dai vincoli che ci sono tra le due braccia, infatti, se a livello teorico si possono ottenere configurazioni particolari come quella vista in figura 3.12 nella pratica non è possibile muovere né manualmente né automaticamente i giunti per ottenere quei casi. Possiamo andare quindi a rappresentare gli angoli di movimentazione mediante la seguente immagine:

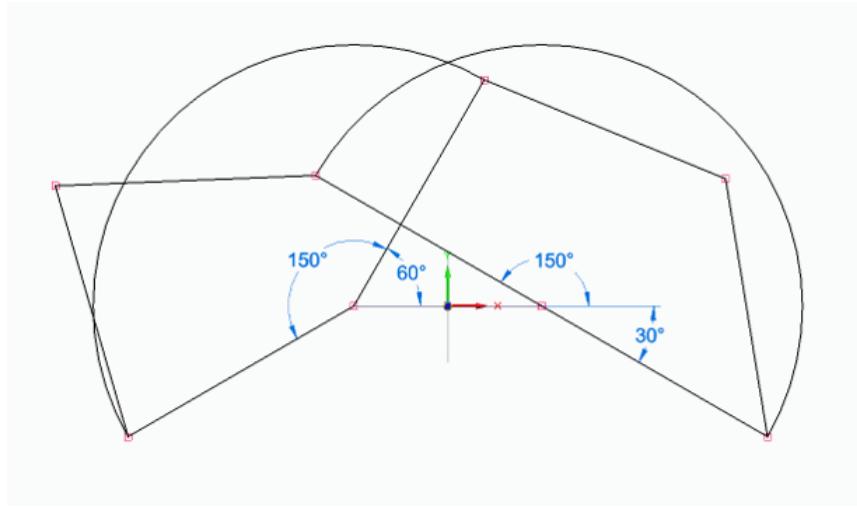


Figura 3.15: Angoli di movimentazione dei giunti motorizzati

Andando ora ad unire gli angoli di movimentazione del manipolatore ed i punti di singolarità, è possibile descrivere lo spazio di lavoro mediante un rettangolo che non viola alcuna condizione

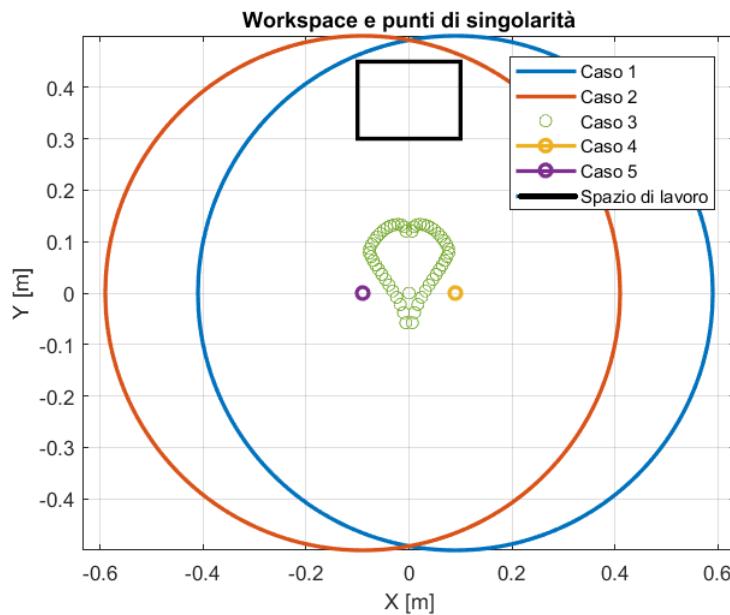


Figura 3.16: *Workspace 5R*

Successivamente, per verificare che lo spazio di lavoro fosse corretto e rispettasse tutte le condizioni è stata fatta un'analisi sul numero di condizionamento del *workspace*:

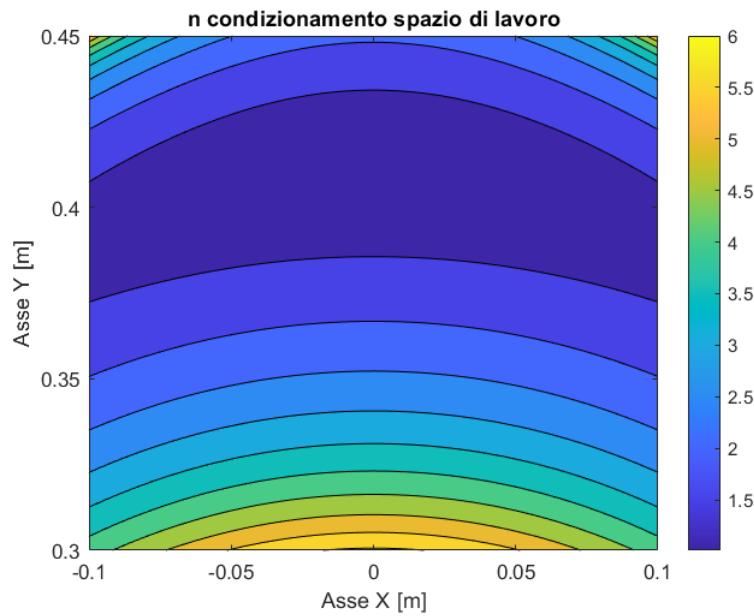


Figura 3.17: Numero di condizionamento workspace

Su tutto il piano x,y si nota come il numero di condizionamento assume valori bassi, questo implica che non vi è singolarità, gli unici punti critici sono quelli esterni, nei quali il valore è vicino a 6, (comunque minore di  $\infty$ ), questi punti vanno a rappresentare casi nei quali il manipolatore avrà più difficoltà a muoversi, ma non sono veri e propri casi di singolarità.

---

## 4 Modellazione end-effector

L'obiettivo di questo capito consiste nel presentare la modellazione della cinematica e dinamica dell'end-effector.

L'end-effector, identificato anche come utensile, è composto da due parti: La prima

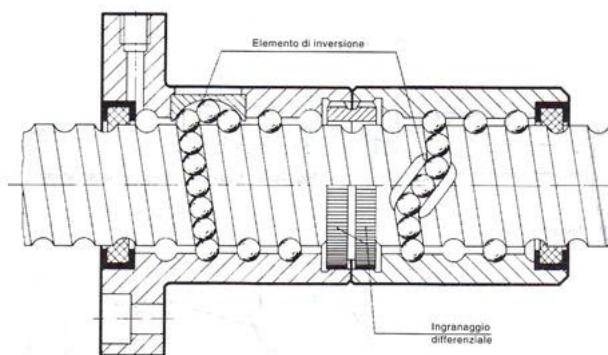


Figura 4.1: Struttura vite

è una vite a ricircolo di sfere che permette un movimento rotativo, mentre la seconda parte è una guida lineare che permette una traslazione positiva o negativa nell'asse z. Entrambi i componenti sono collegati a motori che permettono la movimentazione interfacciandosi con degli azionamenti. All'estremità della vite è possibile collegare un utensile, come è possibile vedere dalla foto è stato collegato un utensile utilizzato per disegnare le traiettorie su di un foglio.



Figura 4.2: Utensile da disegno

Per proseguire con la trattazione della parte cinematica e dinamica andiamo ad introdurre i parametri della vite:

Nome	Descrizione	Valore
$m_v[kg]$	massa vite	0.36
$p_v[m]$	passo vite	0.02
$I_v[kg \cdot m^2]$	momento inerzia vite	$6.40 \cdot 10^{-6}$

Tabella 3: Parametri end-effector

A livello teorico si è partito definendo una legge di moto per entrambi i componenti della vite, si è selezionata una legge polinomiale, per la guida è stata fatta una legge sulla posizione, mentre per la vite a ricircolo di sfere sull'orientamento.

## 4.1 Cinematica end-effector

Come anticipato, abbiamo quindi la presenza di due leggi di moto che identifichiamo con  $z_{ee}$  e  $\varphi_v$ . Il risultato della cinematica di posizione è il parametro  $V$ , per velocità ed accelerazione saranno le sue derivate,  $\dot{V}$  e  $\ddot{V}$  definite come:

$$V = \begin{bmatrix} Z \\ \theta_Z \end{bmatrix}, \dot{V} = \begin{bmatrix} \dot{Z} \\ \dot{\theta}_Z \end{bmatrix}, \ddot{V} = \begin{bmatrix} \ddot{Z} \\ \ddot{\theta}_Z \end{bmatrix}$$

$V$  è composto da una parte di traslazione ( $Z$ ) e da una parte di rotazione  $\theta_Z$ ; Per andar a ricavare la cinematica abbiamo quindi bisogno della legge di moto e della jacobiana della vite, che possiamo definire come:

$$J_e = \begin{bmatrix} \frac{p_v}{2\pi} & \frac{p_v}{2\pi} \\ 0 & 1 \end{bmatrix} \quad (4.1)$$

Andando ora a combinare i parametri e sostituendoli nell'equazione precedente otteniamo:

$$V = J_e \begin{bmatrix} z_{ee} \\ \varphi_v \end{bmatrix}, \dot{V} = J_e \begin{bmatrix} \dot{z}_{ee} \\ \dot{\varphi}_v \end{bmatrix}, \ddot{V} = J_e \begin{bmatrix} \ddot{z}_{ee} \\ \ddot{\varphi}_v \end{bmatrix} \quad (4.2)$$

## 4.2 Dinamica della vite

Per la definizione della dinamica della vite è stata usata, come per la dinamica dei link la tecnica del PLV. Per questa soluzione avremo bisogno delle accelerazioni dei due elementi quindi  $\ddot{Z}_{ee}$  e  $\ddot{\varphi}_v$ . Andiamo poi ad introdurre la matrice di massa della vite, composta dalla massa e dall'inerzia e definita come:

$$M_v \begin{bmatrix} m_e & 0 \\ 0 & I_v \end{bmatrix} \quad (4.3)$$

La soluzione della dinamica la troviamo quindi come:

$$C_{ee} = M_v J_e \begin{bmatrix} \ddot{z}_{ee} \\ \ddot{\varphi}_v \end{bmatrix} \quad (4.4)$$

Sono poi stati eseguiti vari test per la movimentazione della vite a livello teorico, nella seguente immagine è possibile vedere un risultato di questi test, in particolare nella figura sono mostrate le coppie fornite ai motori della vite e la posizione iniziale e finale della vite, tutto questo in funzione del tempo.

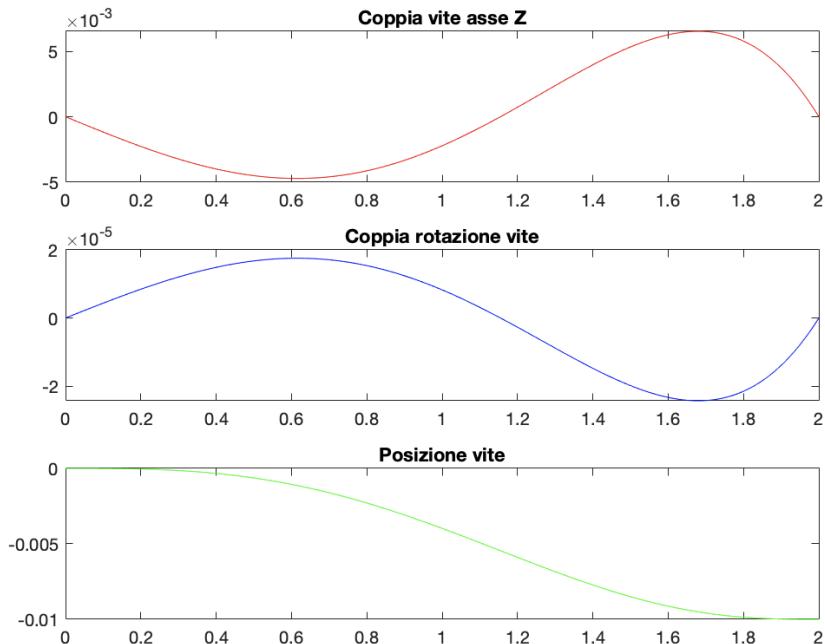


Figura 4.3: Coppie e posizione end-effector

---

## 5 Tecnologie implementate

Conclusa la spiegazione della modellazione di bracci e vite è il momento di iniziare l'attività sperimentale. Lo scopo di questa sezione è l'introduzione alle tecnologie software utilizzate per la modellazione fisica del manipolatore.

### 5.1 Simulink Real time

Simulink real-time è un plugin di matlab, consente di creare applicazioni real-time ed eseguirle su un hardware target, come per esempio un computer. Simulink-real time gira su un computer di "sviluppo" che è il PC utente, ed è diverso dal sistema che permette il movimento in real-time e l'esecuzione del modello reale ovvero il dispositivo target. Di conseguenza simulink real-time può raggiungere tempi di campionamento molto più veloci per uno stesso modello rispetto ad kernel real-time che condivide le risorse hardware con il computer, vi è la possibilità di raggiungere anche i 20 kHz. Un'altra caratteristica fondamentale è il fatto che su SLRT è possibile far funzionare il dispositivo target in modalità *stand-alone* cioè può riuscire ad avviarsi senza il bisogno di una procedura di installazione.

### 5.2 EtherCAT

L'obiettivo di questa sottosezione è quello di presentare il protocollo di comunicazione utilizzato per interfacciarsi col robot PKM, evidenziandone le sue caratteristiche principali.

EtherCAT è una tecnologia ethernet sviluppata in origine da Beckhoff automation, il protocollo è stato pubblicato nello standard IEC61158<sup>4</sup>, soddisfa requisiti *hard* e *soft* real time in particolare nell'ambito dell'automazione. Una caratteristica particolare sono i tempi di ciclo che sono molto veloci infatti una durata media di un tempo di ciclo è inferiore a  $100\mu s$ . Il principio base di funzionamento si basa sul concetto di *Master/Slave*.

Venne introdotto nell'aprile 2003, e nei mesi successivi è nata una società chia-

---

<sup>4</sup><https://webstore.iec.ch/publication/59890>

mata EtherCAT Technology Group (ETG) che è diventata una delle più grandi organizzazioni ethernet al mondo.

### 5.2.1 Proprietà

Il master EtherCAT invia un pacchetto, chiamato telegramma che va ad attraversare tutti i nodi, ogni singolo slave collegato legge i dati che riguardano lui e scrive i dati prodotti intanto che il telegramma si propaga sulla rete verso i nodi successivi. Non appena il pacchetto arriva all'ultimo nodo, è quest'ultimo che si occupa di reinviarlo al master grazie alla comunicazione full-duplex presa da Ethernet, facendo questo, il flusso di dati teorico riesce a superare i  $100\text{ Mbit/s}$ . Il fatto che il master sia l'unico nodo che può inviare frame in maniera attiva garantisce prestazioni deterministiche. Il master utilizza un Media Access Controller (MAC) standard, senza alcun processore dedicato alla comunicazione. Questo consente di implementare un dispositivo master su qualunque piattaforma hardware dotata di una porta di rete, indipendentemente dal Sistema Operativo o software applicativo utilizzato. I dispositivi EtherCAT slave integrano un cosiddetto EtherCAT Slave Controller (ESC) in grado di processare i frame on-the-fly e in modo puramente hardware, il che rende le prestazioni della rete predicibili e indipendenti dalla particolare implementazione dei dispositivi slave.

### 5.2.2 Gestione della rete

La rete EtherCAT è sicura, viene implementato una tecnologia denominata *safety over EtherCAT* che consente la realizzazione di architetture di sicurezza più semplici e flessibili di una logica standard a relè. Vi è la possibilità di trovare questa tecnologia standardizzata nella specifica IEC 61784-3, il sistema di comunicazione è parte del *black channel*, ovvero una parte considerata non rilevante ai fini della sicurezza; questo fa uso di un solo canale per trasferire sia i dati standard che quelli di sicurezza, i frame di sicurezza vengono identificati come *safety container*, e contengono i dati critici del processo e l'informazione necessaria per garantirne l'integrità. Un *safety container* viene mappato dentro i dati di processo ciclici di comunicazione, possono viaggiare tramite cavi di rame, fibre ottiche e connessioni *wireless*, questo

introduce flessibilità, e rende più semplice e sicuro connettere parti della macchina anche lontane fra di loro. In una macchina completamente connessa, andare ad implementare una funzione d’arresto di emergenza totale risulterà semplice anche se altre parti della macchina sono connesse con tecnologie diverse.

### 5.2.3 Implementazione interfacce

Come abbiamo accennato precedentemente, il principio di funzionamento di *EtherCAT* è il concetto di *master/slave*. Per come si è evoluta questa tecnologia, e considerando che l’interfaccia non richiede una CPU ad elevate prestazioni, è possibile andare ad aggiungere un dispositivo di I/O ad un controllore senza andare ad aumentare significativamente i costi complessivi. Per un cliente, è importante l’interoperabilità tra i dispositivi di più fornitori, per questo prima di poter introdurre un dispositivo sul mercato vengono fatti tutti i test, che verificano che l’implementazione rispetti la specifica EtherCAT.

L’interfaccia master ha l’unico requisito di avere una porta ethernet, per l’implementazione viene utilizzato o l’ethernet controller integrato oppure una schede di rete base; nella maggior parte dei casi l’ethernet controller viene integrato mediante un DMA (Direct Memory Access), in questo modo per l’invio dei dati tra il master e la rete non vengono utilizzate le risorse della CPU. Gli slave scrivono i dati prodotti e leggono quelli a loro indirizzati mentre il telegramma li attraversa, facendo così al master arriva l’immagine già ordinata correttamente, la CPU quindi non è più responsabile dell’ordinamento. I dispositivi *slave* invece utilizzano ESC (*Ethercat slave controller*), solitamente di costo contenuto oppure integrato in un microcontrollore standard, esistono slave semplici che non richiedono nemmeno la presenza di un microcontrollore per il fatto che gli ingressi e le uscite digitali possono essere direttamente collegati all’ESC, mentre per quelli più complessi viene usato un controllore a 8-bit.

## 5.3 CME2

CME2 è un software prodotto da Copley control, e serve per la configurazione degli azionamenti. Le funzioni principali riguardano *auto-phasing* e *auto-tuning*, che van-

no a semplificare la realizzazione del sistema. Oltre a queste due funzioni principali abbiamo anche le tabelle Cam che forniscono un buon approccio per la produzione di movimenti sincronizzati e ripetitivi ad un dispositivo esterno, e la possibilità di definire sequenze fino a 32 indici o sequenze indicizzate. Sono anche predisposte funzioni per l'analisi degli strumenti, la configurazione dei motori dei filtri e dei guadagni, è possibile anche andare a cambiare la modalità operativa, scegliendo quindi se lavorare con un anello in posizione, corrente o coppia. Il collegamento con gli azionamenti è avvenuto tramite cavo ethernet, come si può notare dalle foto gli

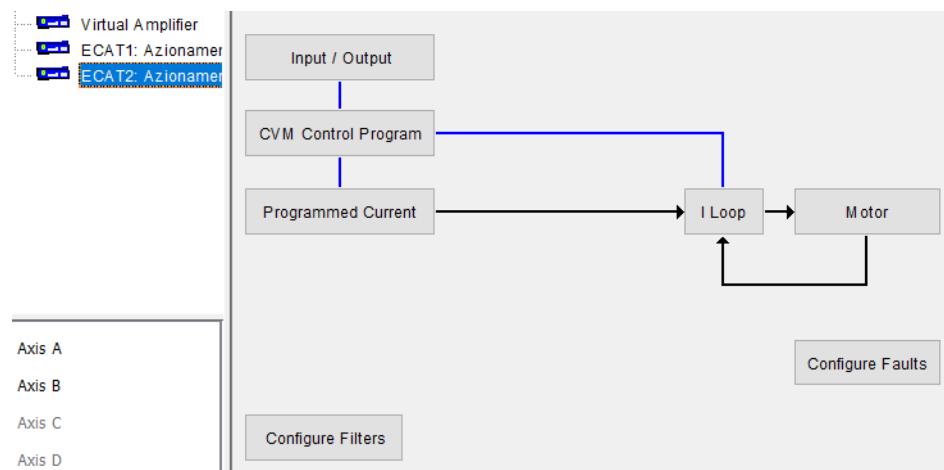


Figura 5.1: Schema azionamento

azionamenti sono controllati in corrente. Un'operazione fondamentale è stata l'analisi dei registri nella quale si è visto a cosa corrispondeva ogni registro. In particolare i registri osservati sono stati quelli dei finecorsa. Per le braccia i registri da guardare sono stati l'ottavo e il quindicesimo, in particolare per vincoli di progetto i finecorsa osservati sono stati quelli al lato destro, per la vite invece il registro osservato è stato il quindicesimo, ed era riferito alla movimentazione superiore della vite.

I/O Line States																	
Inputs									Outputs								
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Lo	Hi	Hi	Hi	Lo	Hi	Lo	Lo	Hi	Lo	Hi	Hi	Lo	Hi	Lo	Lo	X	X

Figura 5.2: Registri azionamento

## 5.4 EC-Engineer

EC-Engineer è uno strumento software utilizzato per la configurazione, diagnostica e monitoraggio delle reti EtherCAT, è nato con lo scopo di aver tutto il necessario su un solo ambiente. Usando questo programma è possibile generare due tipologie di configurazioni EtherCAT, online oppure offline. Il primo tipo di configurazione viene fatto quando siamo direttamente connessi sulla macchina alla rete EtherCAT quindi è un'operazione fatta in real-time, la sonda tipologia invece può essere fatta in laboratorio o ufficio e non richiede la connessione alla rete in quel preciso momento. Per andare a fare una connessione online non è necessario che gli slave siano direttamente connessi al PC locale, questo grazie alla funzione *bus scan* che permette di andare a determinare la topologia della rete facilmente. Nel nostro caso abbiamo scelto una configurazione *offline*, come metodo di comunicazione si è scelto di usare le PDO<sup>5</sup>. Per concludere la configurazione viene esportato un file ENI, ovvero un file XML che descrive la topologia della rete, il comando di inizializzazione per ogni dispositivo ed i comandi che devono essere inviati ciclicamente, il file ENI viene fornito al master che invia i comandi in base a questo file. Una volta creato è stato inserito su simulink nell'*EtherCAT init*.

---

<sup>5</sup>Process Data Object, ovvero dati trasmessi dal/al MotionController, in tempo reale ad ogni nodo ad ogni tempo di campionamento.

---

## 6 Sistema reale

Avendo introdotto anche le tecnologie implementate, andiamo ora a discutere del sistema reale, in particolare concentrandoci sulla sua struttura, compresa la configurazione, le operazioni che hanno consentito la movimentazione e le tecniche di controllo

### 6.1 Struttura del robot

Il manipolatore PKM è un manipolatore a cinematica parallela, composto da due braccia ed un end-effector. Alle braccia sono collegati due motori, uno per il link motorizzato sinistro e l'altro per il link motorizzato destro, i link distali si muovono in conseguenza al movimento di quelli motorizzati. Anche l'end-effector è composto da due motori, il primo motore permette di far salire/scendere la vite, il secondo invece genera un moto elicoidale che permette la rotazione della vite con conseguente salita/discesa. Per quanto riguarda la parte elettronica abbiamo la presenza di due azionamenti che sono collegati uno ai motori delle braccia e l'altro ai motori della vite ed un modulo beckhoff che si occupa della gestione degli input digitali. Per



Figura 6.1: Banco di test

funzionare il sistema ha bisogno di due alimentatori, uno che serve ad alimentare la logica, alimentato a 24 Volt e un altro che serve ad alimentare i quattro motori, 80 Volt.

### 6.1.1 Azionamenti

Gli azionamenti utilizzati sono gli accelnet plus a 2 assi BE2, sono progettati appositamente per EtherCAT, operano con tensioni da 14 a 90 volt, riescono a fornire in uscita fino a 30A.

Sono predisposti per controllo in posizione, velocità e coppia di motori brushless, per la configurazione utilizzano il software CME 2 e la comunicazione avviene mediante l’interfaccia seriale RS-232. Il BE2 opera come ethercat slave, utilizzando il layer applicativo CAN su ethercat CoE. Inoltre, viene fornito un input AuxHV che permette in casi critici di tener vivo l’azionamento anche quando non c’è alimentazione senza perdere le informazioni sulla posizione o le comunicazioni con il sistema di controllo. Per la comunicazione con ethercat invece sono predisposti due cavi RJ-45, la porta d’ingresso IN permette la connessione ad un master o alla porta d’uscita OUT di un dispositivo che nella gerarchia è interposto tra il master e l’azionamento. Inoltre, se l’accelnet è l’ultimo nodo della rete non vi è bisogno di un terminatore sulla porta d’uscita.

### 6.1.2 Beckhoff EK1814

Il beckhoff EK1814 è un accoppiatore EtherCAT che fa da *link* tra il protocollo EtherCAT a livello di bus di campo e il terminali EtherCAT. Inoltre, su questo modello sono anche integrati quattro input digitali e quattro output digitali. La sua struttura lo rende ideale per applicazioni con pochi input/output. L’accoppiatore converte i telegrammi che passano da Ethernet *100BASE-TX* a rappresentazioni di segnali *E-bus*. Una stazione EtherCAT è formata da un accoppiatore e da un numero N di terminali che vengono identificati automaticamente.

Inoltre, l’EK1814 ha due connessioni RJ45, l’interfaccia Ethernet superiore è utilizzata per collegare l’accoppiatore alla rete, mentre quella posteriore serve per il collegamento di altri dispositivi EtherCAT nello stesso commento. Nel nostro pro-

getto è stato usato come master, a questo sono stati connessi gli slave (ovvero gli azionamenti), inoltre gli input e output digitali sono stati usati per controllare la pressione del fungo di emergenza e le luci di segnalazione delle fasi del manipolatore.

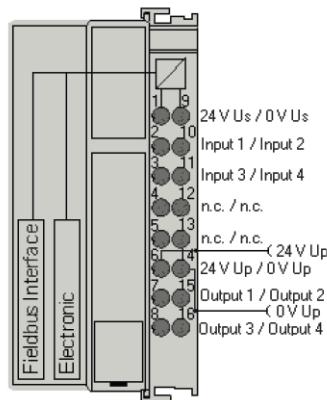


Figura 6.2: Schema modulo bechoff

### 6.1.3 Configurazione della rete

La configurazione della rete prevede alla base il PC Target, in questo vi è una chiavetta USB che fa *runnare* sul pc un sistema operativo simulink real time. Il target è il master della rete, ha due uscite ethernet, la prima è collegata direttamente al modulo bechhoff, il quale prende l'identità di primo slave, e come abbiamo visto precedentemente, al bechhoff sono attaccati e i due azionamenti che si comportano come slave aggiuntivi slave. Invece, alla seconda porta ethernet, vi è collegato il

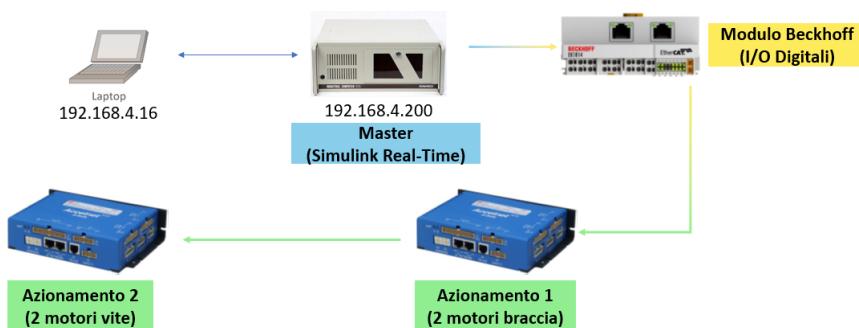


Figura 6.3: Topologia della rete

## 6.1 Struttura del robot

---

PC dell'utente, il quale provvede a generare, compilare, e caricare ed i programmi sul PC target. Da User-PC è anche possibile vedere i grafici e fare delle analisi sui movimenti e le traiettorie eseguite dal manipolatore. La connessione avviene tramite una rete ethernet, l'indirizzo del target è 192.168.4.200, invece per User-PC: Va precisato che il pc dell'utente non fa parte della rete ethercat, ma la rete inizia

```
Ethernet adapter Ethernet 3:  
Connection-specific DNS Suffix . :  
Link-local IPv6 Address . . . . . : fe80::61ef:4e4:4849:4a71%10  
IPv4 Address . . . . . : 192.168.4.16  
Subnet Mask . . . . . : 255.255.255.0  
Default Gateway . . . . . :
```

Figura 6.4: Configurazione rete ethernet user PC

soltanto dal pc target in poi, infatti, ad esclusione delle operazioni viste prima il manipolatore non ha bisogno del pc utente per funzionare. Una volta configurata la

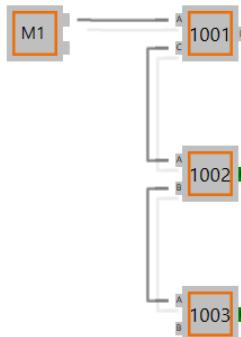


Figura 6.5: Topologia rete mediante Ec-engineer

rete, il passo successivo è stato quello della configurazione dei messaggi, come è stato anticipato nel capitolo precedente il metodo di comunicazione sono le PDO. Le PDO possono essere in input o in output, la differenza sta nel fatto che le primo sono PDO che gli azionamenti trasmettono al master, di conseguenza il master le riceve, quelle di output invece sono PDO che il master trasmette e che gli azionamenti ricevono.

Nelle immagini seguenti sono elencate le PDO di input, in particolare:

- PDO1 e PDO2 contengono i parametri di posizione, velocità coppia effettiva e modalità operativa che vengono trasmesse dagli azionamenti

- PDO3, contiene input generici che sono indipendenti dalla modalità operativa del motore, in particolare è presente il *general purpose inputs* che è il registro che permette la visione dei finecorsa
- PDO4 contiene *status word* e *control word*, sono registri importanti che servono per verificare la modalità operativa e lo stato dell'azionamento, quindi sono utili per capire se l'azionamento è in fase pre-operativa, operativa o in errore

Transmit PDO 1		
Name	Index	Bit Length
Axis A Modes of operation	0x6060:00	8
Axis A Actual motor position	0x6064:00	32
Axis A Actual motor velocity	0x606C:00	32
Axis A Torque actual value	0x6077:00	16

(a) PDO Input 1

Transmit PDO 2		
Name	Index	Bit Length
Axis B Modes of operation	0x6860:00	8
Axis B Actual motor position	0x2A40:00	32
Axis B Actual motor velocity	0x6869:00	32
Axis B Torque actual value	0x6877:00	16

(b) PDO Input 2

Transmit PDO 3		
Name	Index	Bit Length
General purpose inputs	0x2190:00	16
Axis A Peak current limit	0x2110:00	16
Axis B Peak current limit	0x2910:00	16

(c) PDO Input 3

Transmit PDO 4		
Name	Index	Bit Length
Axis A Status word	0x6041:00	16
Axis B Status word	0x6841:00	16
Axis A Control word	0x6040:00	16
Axis B Control word	0x6840:00	16

(d) PDO Input 4

Figura 6.6: PDO in input

Per quanto riguarda le PDO che riceve l'azionamento sono solo due, ed i parametri ricevuti sono:

- *Modes of operation*, è un registro che specifica la modalità con la verrà controllato l'azionamento, per esempio coppia, posizione, velocità o ciclica
- *Target torque*, specifica il valore di coppia che l'azionamento dovrà fornire al motore

Receive PDO 1		
Name	Index	Bit Length
Axis A Modes of operation	0x6060:00	8
Axis A Target Torque	0x6071:00	16

Receive PDO 2

Receive PDO 2		
Name	Index	Bit Length
Axis B Modes of operation	0x6860:00	8
Axis B Target Torque	0x6871:00	16

Figura 6.7: PDO Output 1 e 2

## 6.2 Implementazione nel sistema reale

Una volta ottenuto il file ENI contenente la topologia della rete è stato utilizzato simulink real-time per implementare la logica di controllo del manipolatore Il

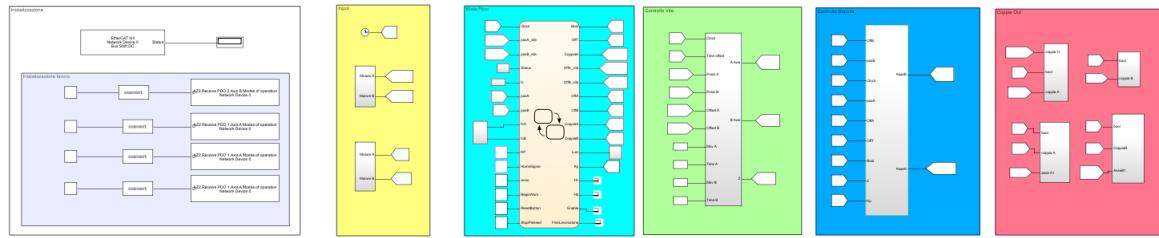


Figura 6.8: Schema generale simulink

programma è stato diviso in sei stati diversi, andiamo ad analizzare ora i vari stati.

### Inizializzazione

La prima fase è quella di inizializzazione, in questa fase viene inserito il file ENI e viene specificata la modalità operativa degli azionamenti. Oltre al file ENI viene

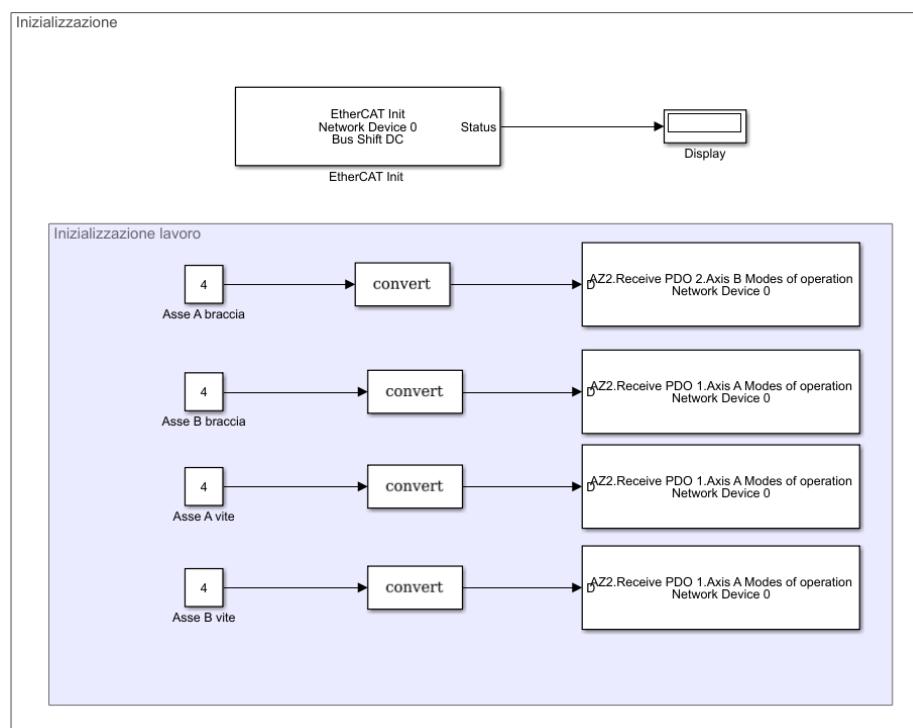


Figura 6.9: Fase 1: Inizializzazione

anche specificata la porta di comunicazione ed il bus che verranno utilizzati per lo scambio di dati. Ogni azionamento ha poi una determinata modalità operativa, come abbiamo visto nelle sezioni precedenti il controllo è effettuato in coppia.

Modalità	Descrizione
1	modalità profilo in posizione
3	modalità profilo in velocità
4	modalità profilo in coppia
6	modalità homing
7	modalità posizione interpolata

Tabella 4: Tipologie controllo azionamenti

## Input

La seconda fase è quella di input, in questa fase andiamo a prendere tutti i valori di posizione dei motori sia della vite che delle braccia I valori vengono presi mediante

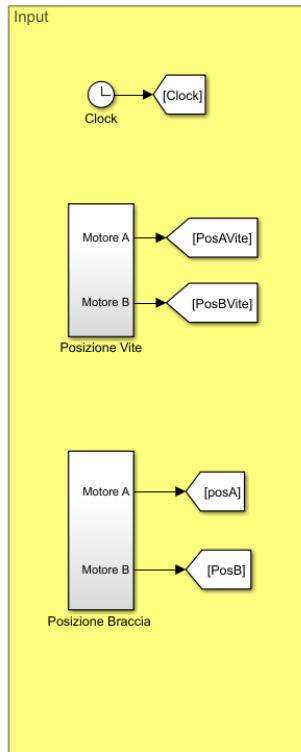


Figura 6.10: Fase 2: Input

i messaggi dalle PDO ed hanno bisogno di essere convertiti, proprio per questo la struttura di ricezione di un messaggio è la seguente:

## 6.2 Implementazione nel sistema reale

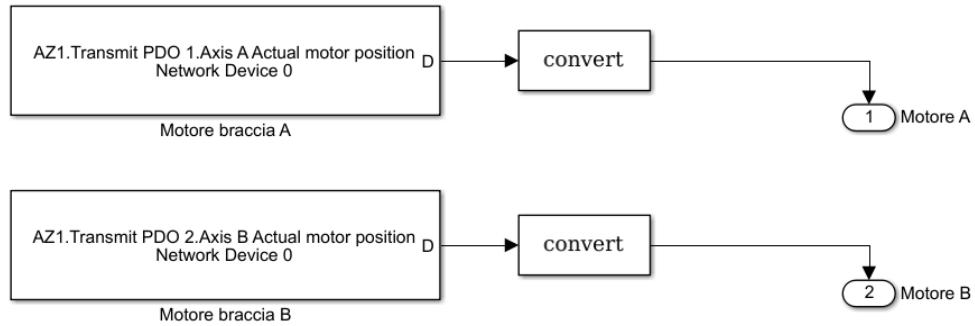


Figura 6.11: Conversione e lettura motori

### Stateflow

Lo stateflow è un blocco dove viene posta la logica fondamentale dell'applicazione, in particolare in input avremo tutti i dati come ad esempio il clock, lo stato dei finecorsa, le posizioni dei motori e degli input che serviranno per l'interfaccia grafica. In output invece abbiamo gli offset, che ci serviranno per capire di quanto ci siamo

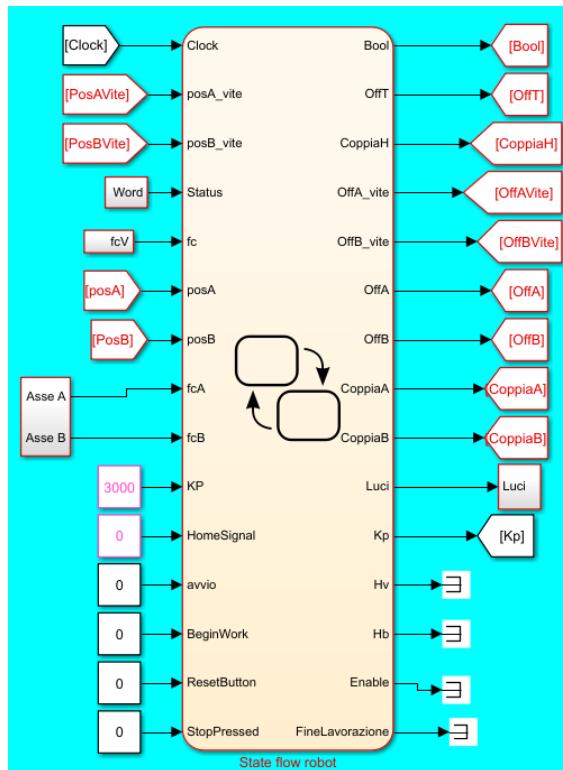


Figura 6.12: Fase 3: Stateflow

mossi nelle varie fasi, e quindi avere un riferimento sia di posizione che temporale,

le coppie di homing dei motori e della vite, e un blocco relativo alla gestione delle luci<sup>6</sup>.

### Controllo vite

Il blocco Controllo Vite, contiene lo schema del controllore implementato per controllare la vite, in ingresso abbiamo il clock, le posizioni dei motori con relativi offset e la movimentazione da far eseguire ad entrambi i motori della vite con il tempo di esecuzione. Grazie a tutti questi parametri potremo definire le leggi di moto che

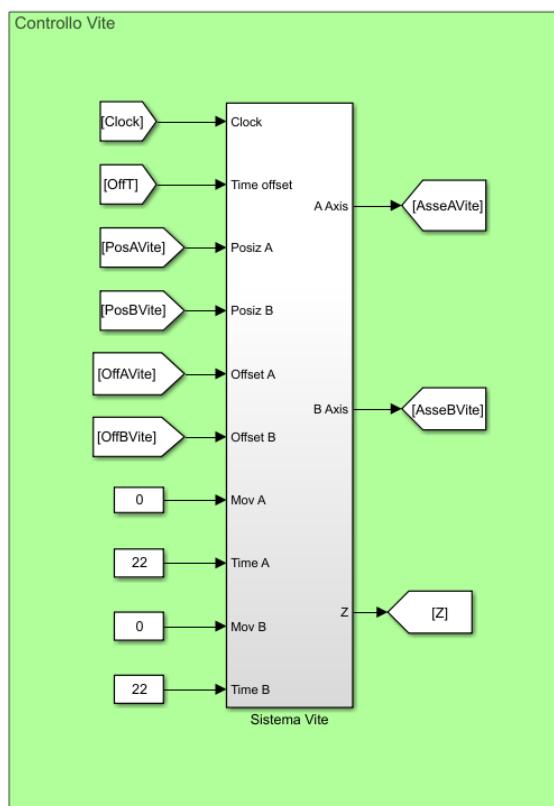


Figura 6.13: Fase 4: Schema di controllo della vite

ci permetteranno il movimento nell'asse Z. Importante è sapere che il movimento eseguito dalla vite a ricircolo di sfere è dipendente anche dal movimento eseguito dalla guida lineare, infatti la rotazione provoca anche un abbassamento della vite, per risolvere questo problema il motore della guida dovrà essere sempre pronto a

<sup>6</sup>abbiamo la presenza di tre luci: rosso, bianco e verde, nei paragrafi successivi verrà introdotto il loro comportamento

rispondere e correggere questa situazione; in caso che i due motori vadano alla stessa velocità si avrà una rotazione senza traslazione.

### Controllo braccia

Il blocco Controllo Braccia è quello responsabile della movimentazione dei link motorizzati, come per il blocco della vite prende in ingresso le posizioni, il *clock* con i relativi offset e dentro vengono svolte le operazioni di generazione della legge di moto e di controllo. In uscita avremo le coppie che saranno assegnate ai due assi.

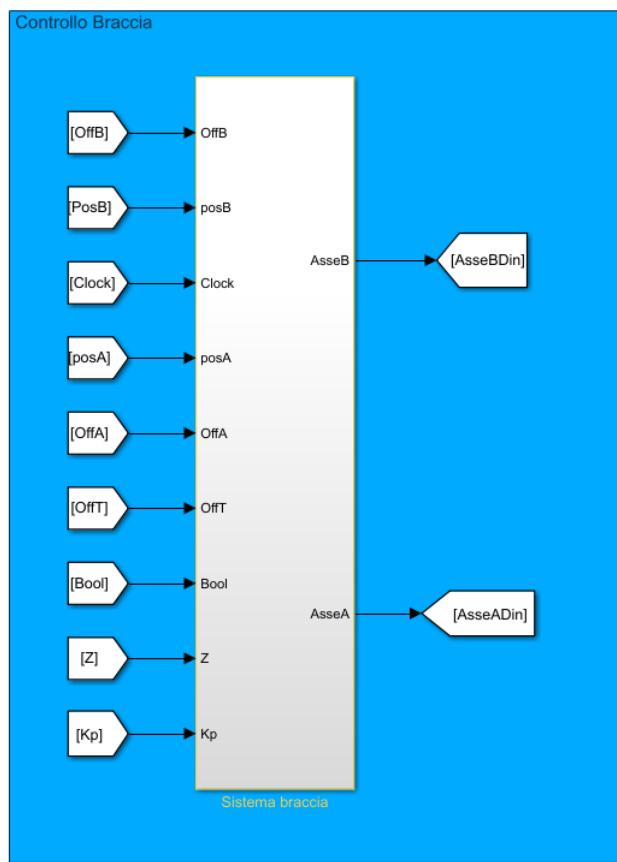


Figura 6.14: Fase 5: Schema di controllo delle braccia

### Coppie uscita

Dopo aver ottenuto le coppie di homing dallo stateflow e le coppie dei motori dagli schemi di controllo è venuto il momento di inviare le coppie agli azionamenti e di conseguenza ai motori; per far questo usiamo un blocco per ogni motore, oltre alle

coppie in entrata avremo anche una variabile di controllo. Nella figura precedente

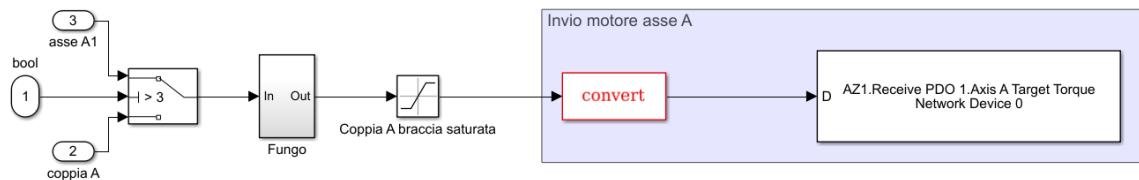


Figura 6.15: Schema espanso invio coppia

è possibile vedere lo schema espanso come spiegato, lo switch permette la scelta in base alla variabile bool, fintanto che è minore o uguale a 3 verrà erogata solo la coppia di Homing, quando arriva a 4 invece vuol dire che siamo nella fase di controllo, di conseguenza verrà erogata la coppia di controllo. Successivo allo switch c'è un blocco che serve per la gestione delle emergenze, infatti, una volta premuto il fungo verrà assegnata una coppia costante uguale a 0 che fermerà la lavorazione ed anche dopo che verrà sbloccato il fungo la coppia per sicurezza rimarrà a zero, l'unico modo per resettare questa condizione è il riavvio del programma.

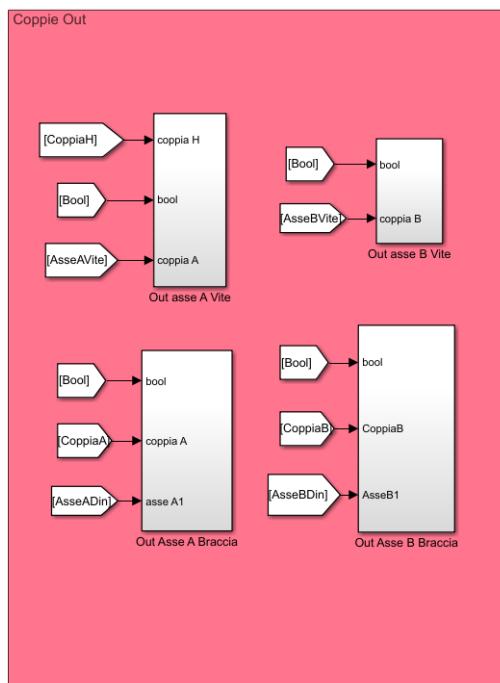


Figura 6.16: Fase 6: Copie in uscita

Successivo al blocco fungo abbiamo un saturatore, questo serve per evitare di dan-

neggiare il manipolatore in caso che le coppie computate siano molto alte, è stato trovato sperimentalmente un limite che coincide con la coppia nominale che non può essere superato, per concludere l'ultima parte è quella che si occupa di inviare mediante PDO il valore di coppia convertito all'azionamento.

Andiamo ora a concentrarci sulle parti principali di questo programma, in particolare andremo a trattare lo stateflow il metodo di funzionamento ed i vari stati, passeremo poi all'interfaccia grafica che permette di comandare il manipolatore, e per concludere andremo a vedere i controllori implementati per la vite e per le braccia, andando a vedere la struttura, lo schema e i risultati ottenuti per ogni approccio.

## 6.3 Stateflow

*Stateflow* si occupa di fornire diagrammi di transizione, stato e di flusso utilizzando un linguaggio grafico. Nel caso del manipolatore è stato utilizzato per la progettazione di diagrammi di transizione in base agli stati del robot. In questa sezione andremo a vedere le fasi gli stadi di evoluzione che sono stati costruiti.

### 6.3.1 Fase di Homing

Appena il robot viene acceso non possiamo sapere dove si trova, di conseguenza abbiamo bisogno di uno stadio che ci vada a trovare una posizione di riferimento nella quale sappiamo dove è collocato effettivamente. Il primo stadio è quindi quello di *homing*, consiste nel portare i motori a toccare i finecorsa indicandogli che quello è il loro punto di partenza. I motori utilizzati per questa fase sono stati quelli delle braccia e quello di traslazione della vite.

L'approccio iniziale è stato quello di fornire una coppia costante che in automatico si occupava di andare a toccare i finercorsa, e dopo che li toccava si passava nello stato successivo. Però, per motivi di sicurezza e considerando che lasciando fermo il manipolatore per diverso tempo la stessa coppia magari potrebbe non farlo muovere si è deciso di chiudere l'anello in posizione, in particolare per eseguire la fase di homing è stata data una rampa con pendenza negativa in quanto i finecorsa vengono rilevati quando i link sono totalmente a destra. Per controllare questa rampa è stato fatto un controllo proporzionale e integrale sull'errore tra la posizione attuale ed il

riferimento, la legge implementata è quindi del tipo:

$$PI = \frac{K_p s + K_i}{s}$$

Nella figura successiva andiamo a vedere lo schema effettivamente implementato per questa prima fase:

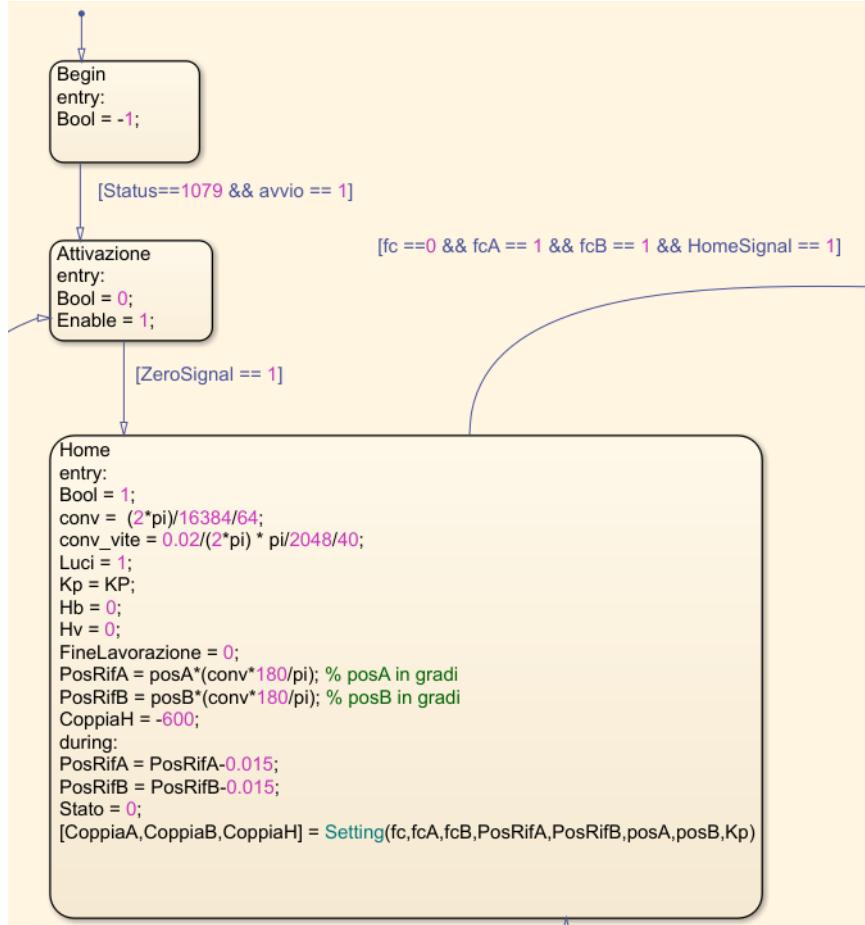


Figura 6.17: Fase di Homing

Il primo stato è quello di begin, appena il programma si avvia ci entriamo in automatico, per passare allo stato successivo, ovvero attivazione abbiamo bisogno di due condizioni, la prima si verifica quando la *Status Word* è uguale a 1079, ovvero quando gli azionamenti sono usciti dalla fase *pre-operational* e sono quindi pronti all'uso, la seconda condizione invece è quando il segnale avvio è vero, nello stato attivazione i motori sono ancora fermi però viene abilitato il loro utilizzo, passare quindi in questo stato è obbligatorio. Per passare allo stato successivo il segnale

*ZeroSignal* deve essere vero, questo è un segnale gestito mediante un bottone da interfaccia grafica, appena viene premuto entriamo nello stato **Home**. In questo stato vengono definite variabili per la conversione dei valori<sup>7</sup>, viene poi salvata la posizione di riferimento dell'asse A e B dei motori delle braccia. La fase successiva è quella del *during*, rimanendo in quello stato quella fase viene eseguita ad ogni ciclo (1ms), in questa abbiamo la rampa che decresce di 0.015 gradi al millisecondo (15 gradi al secondo) e abbiamo una funzione simulink che è quella che si occupa del controllo PI. Una volta raggiunta la posizione del finecorsa le coppie vengono settate a 0, impedendo quindi un'ulteriore movimentazione. Per passare alla fase successiva abbiamo bisogno che tutti gli elementi siano arrivati a finecorsa.

### 6.3.2 Fase di posizionamento

La fase successiva è quella di posizionamento, per non lasciare il robot nella configurazione di zero si è scelto di spostarlo in una configurazione standard lontana dai punti di singolarità e che sarà comoda per le movimentazioni e traiettorie successive. La configurazione scelta prevede che i giunti siano messi a  $100^\circ$  e  $80^\circ$ , anche qua come prima il primo approccio è stato quello di utilizzare una coppia costante per il movimento; la fase di zero lasciava i link a  $60^\circ$  e  $-30^\circ$ , vi era quindi la necessità di fare  $+40^\circ$  per il motore a sinistra e  $+110^\circ$  per il motore a destra, la coppia costante veniva erogata finché non si arrivava a quelle condizioni, dopodiché eravamo sicuri che il posizionamento era stato fatto in maniera corretta. La vite invece ha due modalità di configurazione, è possibile farla abbassare o lasciarla alta in finecorsa, si è scelto di proseguire in questo modo in quanto alla vite come abbiamo visto precedentemente è possibile collegare degli utensili e quindi l'abbassamento la predisponeva al disegno per traiettorie bidimensionali. L'approccio di controllo finale però non è stato quello della coppia costante, anche in questo caso per motivi di sicurezza ma, sapendo le posizioni finali che si volevano raggiungere, si è optato per definire una legge di moto che si occupa di portarci nella condizione di posizionamento desiderata

---

<sup>7</sup>I valori presi dai motori sono espressi tutti in conti, per quello è stata necessaria una fase di analisi dei motori per capire come convertirli, successivamente c'è stato un passaggio da counts a radianti e da radianti a gradi.

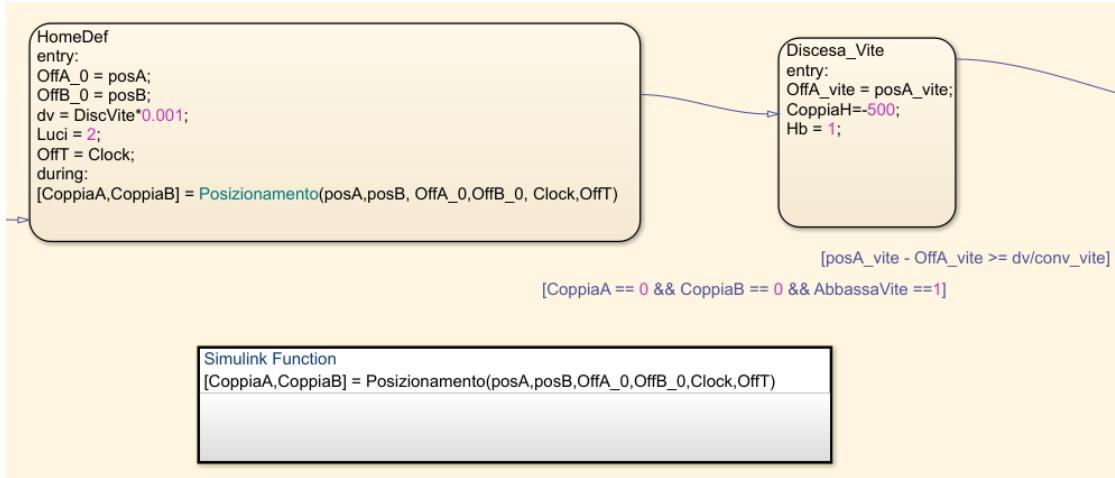


Figura 6.18: Fase di posizionamento

Riprendendo lo schema precedente, per passare allo stato **HomeDef** bisogna aver raggiunto tutti i finecorsa; non appena entriamo vengono salvati gli offset della posizione delle braccia, del tempo e la posizione desiderata di discesa della vite. Ad ogni ciclo, quindi ogni millisecondo viene eseguita la funzione **Posizionamento(...)** la quale, mediante una legge di moto, ci consente di eseguire in maniera corretta il posizionamento. Andando ad analizzare nello specifico il blocco simulink del posizionamento, vengono costruite due leggi polinomiali, una per ogni motore che forniscono un *setpoint* in posizione in gradi, che verrà confrontato con la posizione attuale. Per ottenere poi una coppia, e garantire errore a transitorio esaurito viene introdotto un sistema di controllo proporzionale integrativo, con legge di controllo:

$$\tau = K\tilde{\theta} = \begin{bmatrix} K_p & K_p \\ K_i & K_i \end{bmatrix} \begin{bmatrix} \tilde{\theta}_1 \\ \tilde{\theta}_2 \end{bmatrix} = K_p\theta + K_i\dot{\theta}$$

Le coppie di controllo uscenti andranno quindi al manipolatore e si occuperanno della movimentazione. Possiamo vedere lo schema della legge di moto implementato in figura 6.19.

Una volta eseguito il posizionamento, lo stato successivo riguarderà la discesa della vite, per passarci le coppie dei bracci dovranno essere pari a zero e sull'interfaccia grafica dovrà essere premuto il bottone relativo alla discesa. In particolare per la discesa verrà utilizzata la variabile **dv** salvata precedentemente che indica di

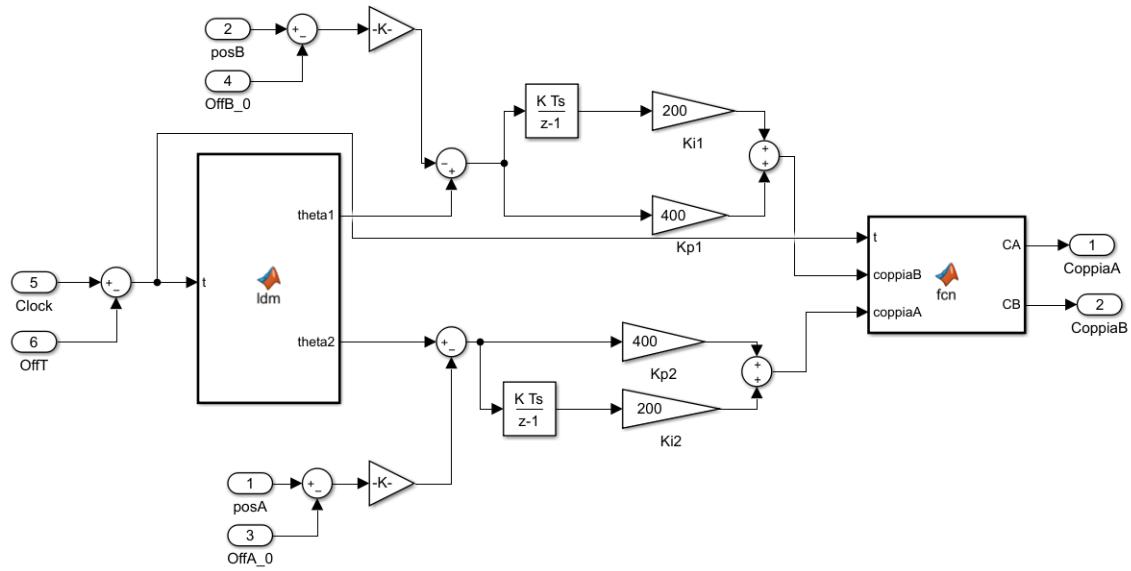


Figura 6.19: Implementazione funzione di posizionamento

quanti centimetri la vite deve scendere; per farla scendere verrà fornita una coppia al motore di traslazione che si occuperà di arrivare alla posizione desiderata.

#### 6.3.3 Fase di controllo

L'ultima fase è quella di controllo ed esecuzione della traiettoria, in questa fase andiamo a settare l'offset dei motori della vite, del tempo e dei motori delle braccia, serve resettare l'offset della vite in quanto il posizionamento (nel caso di discesa) è stato appena concluso; l'offset del tempo ci permette di far partire virtualmente il tempo da zero negli schemi di controllo dopo che siamo arrivati alla configurazione di *Homing*. Come abbiamo anticipato precedentemente nello stato **OffSet** c'è il settaggio degli *offset* e tutte le coppie vengono messe a zero per evitare eventuali movimentazioni indesiderate. Per passare alla fase successiva abbiamo bisogno che venga premuto il bottone **Work** sull'interfaccia grafica, questo garantirà il passaggio allo stato *Controllo*, può darsi che il bottone non venga premuto subito, di conseguenza il clock potrebbe aumentare, per questo l'offset del clock viene settato nella fase di controllo. Per quanto riguarda la scelta della traiettoria tramite interfaccia grafica vi è la possibilità di scegliere quella desiderata. A livello implementativo le traiettorie non sono altro che leggi di moto fatte in due o tre dimensioni, dipenden-

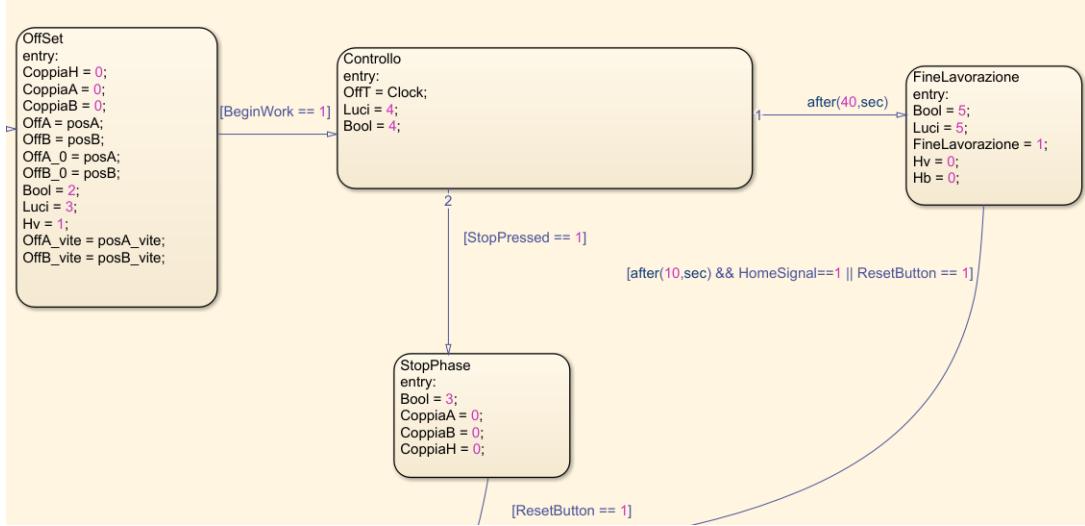


Figura 6.20: Fase di controllo

temente dal caso, dopo che il manipolatore inizia ad eseguire la traiettoria possono esserci due evoluzioni:

- la traiettoria viene eseguita correttamente
- la traiettoria da problemi

Nel primo caso abbiamo 40 secondi per eseguire la traiettoria (il tempo può essere personalizzato in base alla tipologia di traiettorie), alla fine di questo tempo passiamo allo stato **FineLavorazione** dove il manipolatore è fermo ed ha concluso la sua traiettoria, da questo mediante il bottone di reset è possibile ritornare alla fase di zero, oppure in automatico dopo un determinato periodo di tempo se il tasto di homing è stato lasciato attivo il manipolatore torna all'homing. Nel secondo caso ci si accorge che la traiettoria sta dando problemi, magari vibrazioni o si vede che il manipolatore rischia di entrare in singolarità, per prevenire questo vi è un bottone denominato **STOP** che permette l'arresto immediato del manipolatore, azzerando tutte le coppie. A differenza della pressione del fungo, che dopo lo sbloccaggio richiede il riavvio del dispositivo, nel caso in cui si entri nella fase di stop mediante il bottone di reset è possibile far tornare il manipolatore nella fase di attivazione, da questa poi sarà possibile far partire di nuovo la fase di zero e successivamente quella di homing.

### Gestione variabile di stato e luci

Durante tutte le fasi è possibile vedere nello stateflow che due variabili si evolvono costantemente: bool e luci; la prima serve per indicare lo stato nel quale si trova il manipolatore, in particolare è stata definita la seguente tabella degli stati.

Valore	Stato
-1	Pre-operativo
0	Attivo
1	Homing
2	Manipolatore posizionato
3	Traiettoria
4	Manipolatore fermo

Tabella 5: Valori variabile *bool*

La seconda invece serve a pilotare le luci in modo tale da avere un feedback visuale che indica fase in cui è il manipolatore.

Valore	Colore	Bool
1	Bianco	1
2	Bianco Rosso	1
3	Rosso	2
4	Bianco Verde	3
5	Verde	4

Tabella 6: Valori luci

#### 6.3.4 Interfaccia grafica

Per gestire al meglio le varie impostazioni, e per essere sicuri del passaggio tra i vari stati è stata creata un'interfaccia grafica *user-friendly* mediante *instrument panel* fornito da *Simulink real-time explorer*

A sinistra, dall'alto in basso possiamo vedere dei LED che hanno la funzionalità di:

- Fungo: rimane acceso fintantoché il fungo non è premuto
- Homing: si accendono quando i motori hanno raggiunto il finecorsa
- Fine lavoro: appena la traiettoria assegnata finisce questo si accende

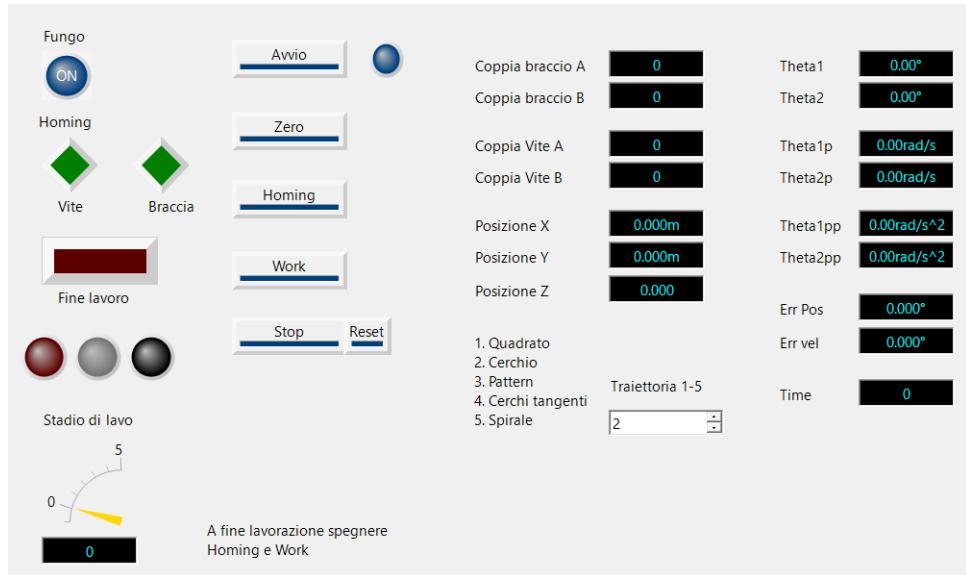


Figura 6.21: Interfaccia grafica

- Luci: sono i led visti nella tabella 6, che vanno ad indicare in che fase di lavoro è il manipolatore, per comodità visiva il led bianco nell’interfaccia è stato sostituito da uno ocra

Sotto questi led abbiamo un indicatore che va a specificare le fasi di lavoro come viste nella tabella 5. Al centro abbiamo i bottoni che permettono di passare tra le varie fasi dello stateflow e di fare tutte le operazioni quindi avvio, zero, homing, lavoro stop e reset. In particolare i bottoni funzionano tramite il collegamento ad una variabile, e quelle di riferimento sono le ultime costanti collegate in input allo stateflow visto in figura 6.12. A destra abbiamo invece una parte di visualizzazione dove vediamo tutti i parametri di interesse del manipolatore, in particolare le coppie fornite e le posizioni sia nel piano  $[x, y, z]$  che quelle ai link motorizzati quindi  $\theta_1, \theta_2$ . Infine, abbiamo il selettori di traiettoria, il quale permette la scelta fra le sei opzioni possibili (di default viene eseguito il cerchio).

1. Quadrato
2. Cerchio
3. Pattern
4. Cerchi tangenti

5. Spirale

6. Solo asse Z

Adesso che abbiamo introdotto la logica di funzionamento a stati, proseguiamo la nostra trattazione andando a vedere le tipologie di controllo che sono state implementate, in particolare inizieremo col guardare il controllo applicato alla vite e successivamente quello per le braccia.

## 6.4 Controllo vite

Lo scopo di questa sottosezione è quello di introdurre le due tipologie di controllo per la vite, da un punto di vista teorico e andare ad analizzare il comportamento pratico. Il sistema, essendo formato da due parti, e potendo fare solo due movimenti è ad un solo grado di libertà non risulta essere quindi molto complesso. Gli approcci di controllo utilizzati sono di tipo centralizzato, l'analisi di questi controllori verrà fatta mediante funzioni di trasferimento, proseguire con andiamo quindi ad introdurre la legge di controllo di un controllore PID generico:

$$\tau_v(t) = K_p \left( q_v(t) + \frac{1}{T_I} \int_0^t q_v(\tau) d\tau + T_D \frac{dq_v(t)}{dt} \right) \quad (6.1)$$

### 6.4.1 Controllo proporzionale

Il primo controllo implementato è quello proporzionale, questa tipologia di controllo si basa sull'idea che ingresso ed uscita siano legati in modo algebrico da un coefficiente  $K_p$  chiamato anche guadagno proporzionale. La legge di controllo è quindi definita come

$$\tau_v = K_p q_v \quad (6.2)$$

L'azione proporzionale è utile in quanto più grande è l'errore all'ingresso del controllore e maggiore sarà l'azione svolta da esso. Il regolatore, riprendendo l'equazione 6.1 si può notare che solo con il contributo proporzionale la velocità di risposta del sistema aumenta, però con guadagno elevato diminuisce la stabilità e quindi au-

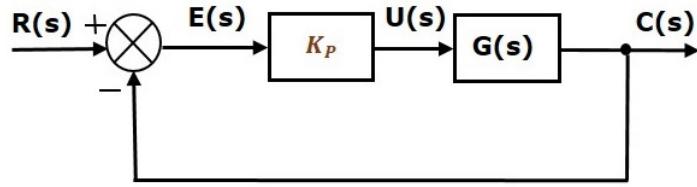


Figura 6.22: Schema teorico controllore proporzionale

mentano le oscillazioni<sup>8</sup>. Andiamo ora a vedere l'implementazione del controllore:  
INSERIRE DESCRIZIONE E ALTRO

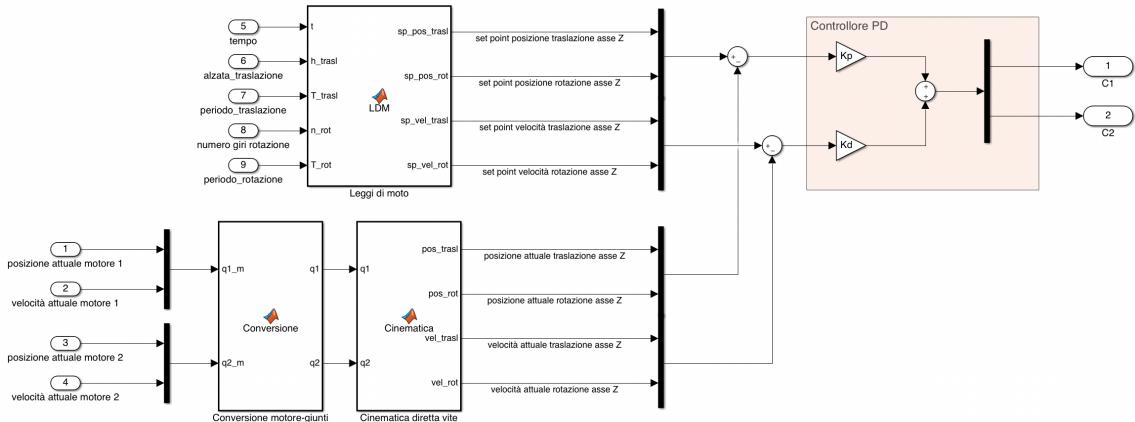


Figura 6.23: CAMBIARE

#### 6.4.2 Controllo proporzionale derivativo

A partire dal controllore precedente andiamo ad aggiungere una parte: l'azione derivativa. In uscita fornisce la derivata rispetto al tempo dell'errore, se dovessimo analizzarla singolarmente potremmo definire una legge del tipo:

$$\tau_v = K_D \dot{q}_v \quad (6.3)$$

Abbiamo la presenza della velocità, infatti il controllore derivativo viene anche chiamato controllore di velocità, il suo comportamento è sostanzialmente diverso da

<sup>8</sup>Il parametro proporzionale viene definito ed analizzato molto in letteratura, nella pratica però è solo un parametro teorico, infatti solitamente ci si riferisce alla banda proporzionale, definita come la variazione minima dell'ingresso che porta l'uscita al valore minimo percentuale.

## 6.4 Controllo vite

quello proporzionale in quanto l'uscita dipende dalla velocità con la quale varia l'errore e riesce a fornire un anticipo di fase. Riprendendo l'equazione 6.1 il parametro che controlla questo anticipo è  $T_D$  infatti nel caso in cui  $K_P = K_I = 0$  andando ad analizzare solo il suo comportamento si nota che la stabilità del sistema peggiora sia aumentando che diminuendo il valore. Andando ora ad unire i due contributi otteniamo un controllore PD. Possiamo scrivere l'azione combinata dei due controllori

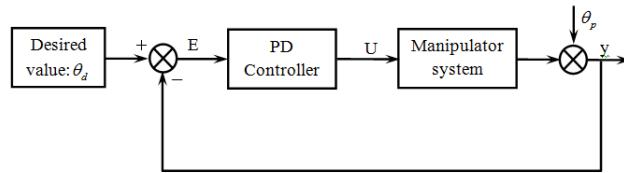


Figura 6.24: Schema teorico controllore PD

considerando il rapporto ingresso uscita e ottenendo:

$$\tau_v = K_P(1 + sT_D) \quad (6.4)$$

con  $T_D = \frac{K_D}{K_P}$ ). In caso di ingresso definito come uno scalino, la presenza del termine derivativo va ad introdurre uno zero e ad aumentare il coefficiente di s, andando a ridurre le oscillazioni e di conseguenza stabilizzando il sistema. Andiamo ora a vedere l'implementazione sul manipolatore del controllore PD: A partire da una

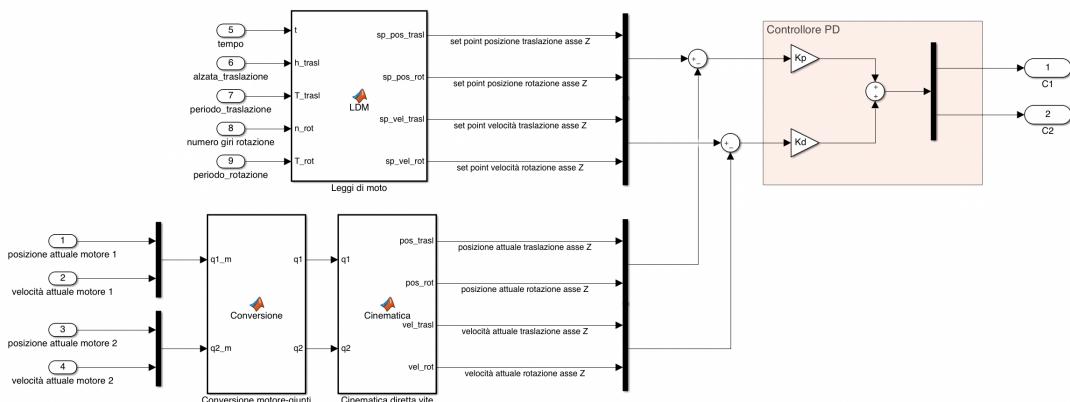


Figura 6.25: Controllore PD Vite

legge di moto per entrambi i motori che fornisce quattro set point, abbiamo: una

prima fase di raccolta dei dati nei quali si prendono le posizioni e velocità dei motori della vite, essendo i motori connessi a riduttori, ed operando sui giunti abbiamo un blocco che si occupa di fare la conversione; una volta fatta la conversione vi è la necessità di effettuare la cinematica diretta per poter passare dagli angoli alle coordinate dell'asse z. In seguito verrà fatta una differenza tra le posizioni e velocità ottenute con la cinematica e quelle del *set-point*; gli errori verranno poi moltiplicati per  $K_P$  (posizione) e  $K_D$  (velocità) per ottenere i valori della legge di controllo da assegnare ai motori.

INSCRISCI CONFRONTI ETC

## 6.5 Controllo braccia

Questa sottosezione si pone come obiettivo l'introduzione di approcci di controllo noti in letteratura e applicarli ai link motorizzati del manipolatore; in particolare andando ad analizzare il loro funzionamento a livello teorico, la loro implementazione pratica ed i risultati ottenuti, cercando quindi il controllore migliore. Tutte le tipologie di controllo introdotte saranno di tipo centralizzato, gli algoritmi di questa tipologia sfruttano una conoscenza più approfondita del modello dinamica del manipolatore in modo tale da compensare i termini di accoppiamento non lineari, verranno quindi introdotte coppie di compensazione per i termini NL, inoltre, qualsiasi incertezza della struttura e qualunque imprecisione nella misura della posizione daranno origine ad una perdita di accuratezza e quindi a problemi di controllo.

### 6.5.1 Controllo proporzionale derivativo

La prima tipologia di controllo applicata è stata quella proporzionale derivativa. L'obiettivo di questa tecnica è quello di risolvere il problema della regolazione, ovvero assegnare la posizione corretta all'end-effector rispetto ad un riferimento di equilibrio costante. Non viene risolto il problema della dinamica con la quale si raggiunge quella configurazione, però come obiettivo ci si pone di trovare la struttura del controllore che ci assicuri una stabilità asintotica in quella specifica posa desiderata. Definiamo lo stato come:

$$\tilde{q} = q_m^0 - q_m \quad (6.5)$$

Con  $q_m^0$  che rappresenta il *set-point* (posizione desiderata) e  $q_m$  la posizione attuale, per poter andare a risolvere il problema abbiamo bisogno di introdurre il metodo diretto di Lyapunov, il quale dice che:

*L'analisi della stabilità di un punto di equilibrio viene fatta utilizzando, oltre alle equazioni di stato del sistema, opportune funzioni scalari, dette funzioni di Lyapunov, definite sullo spazio degli stati.*

Definiamo quindi la funzione come:

$$V(q : m, \tilde{q}) = \frac{1}{2} \dot{q}_m M(q_m) \dot{q}_m + \frac{1}{2} \tilde{q} K_p \tilde{q} > 0 \quad (6.6)$$

Il termine  $K_p$  rappresenta la rigidezza del sistema ed è una matrice ( $n \times n$ ) simmetrica e definita positiva. Considerando che il *set-point* è un termine costante possiamo andare a derivare la funzione V rispetto al tempo ottenendo:

$$\dot{V} = \ddot{q}_m M(q_m) \dot{q}_m + \frac{1}{2} \dot{q}_m \dot{M}(q_m) \dot{q}_m - \dot{q}_m K_p \tilde{q}$$

e sapendo che  $\tau_m = M(q_m) \ddot{q}_m + C(q_m, \dot{q}_m)$ <sup>9</sup> possiamo riscrivere la funzione come

$$\dot{V} = \dot{q}_m [\tau_m - K_p \tilde{q}] \quad (6.7)$$

Se adesso consideriamo la legge di controllo PD definita come:

$$\tau_m = K_p \tilde{q} - K_d \dot{q}_m \quad (6.8)$$

e andiamo a sostituirla nell'equazione 6.7 otteniamo

$$\dot{V} = -\dot{q}_m K_d \dot{q}_m \leq 0$$

---

<sup>9</sup>Nella trattazione di Lyapunov appare anche il termine di compensazione gravitazionale però nel manipolatore analizzato questo termine è costante e pari a zero.

Se  $K_d$  è una matrice definita positiva allora  $\dot{V}$  è semidefinita negativa, la dinamica del sistema è quindi:

$$M(q_m)\ddot{q}_m + C(q_m, \dot{q}_m)\dot{q}_m = K_p\tilde{q} - K_d\dot{q}_m \quad (6.9)$$

In particolare abbiamo che velocità ed accelerazioni sono nulla in corrispondenza di  $\dot{V} = 0$ . Lo schema teorico di controllo è uguale a quello visto in figura 6.24 nella sezione della vite.

Possiamo vedere lo schema di controllo implementato sul manipolatore: Avendo

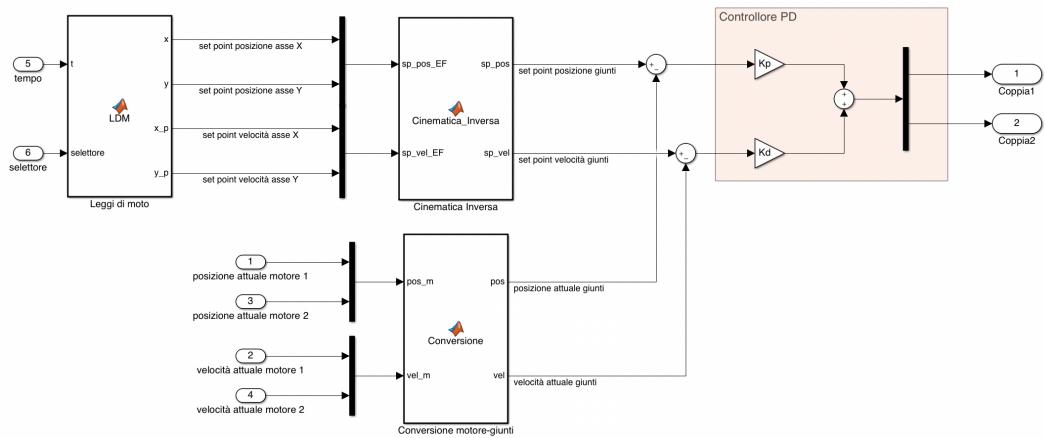


Figura 6.26: Controllore PD braccia

la possibilità di scegliere tra più traiettorie, a partire dal tempo attuale e dalla traiettoria desiderata andiamo a definire una legge di moto che fornirà i *set-point* in posizione, velocità all'end-effector. Per andare a convertire questi in coordinate ai giunti vi è un blocco che si occupa di fare la cinematica inversa. Vengono poi lette le posizioni e velocità attuali dei motori e verranno successivamente convertite ai giunti; adesso che abbiamo tutto a livello di giunti possiamo fare la differenza tra riferimento e reale ottenendo un errore in posizione e uno in velocità<sup>10</sup>. Gli errori verranno moltiplicati per  $K_p$  e  $K_d$  per ottenere la legge di controllo che verrà assegnata ai motori.

Per poter andare a testare il controllore è stata assegnata una legge di moto che disegna un cerchio di raggio 5cm in 5 secondi. Andiamo ora a vedere le coppie

<sup>10</sup>gli errori sono dei vettori [2x1] contenenti le due componenti relative ai due angoli  $\theta_1, \theta_2$ .

## 6.5 Controllo braccia

assegnate ai giunti  $\theta_1$  e  $\theta_2$ : Possiamo poi andare ad analizzare la posizione ottenuta

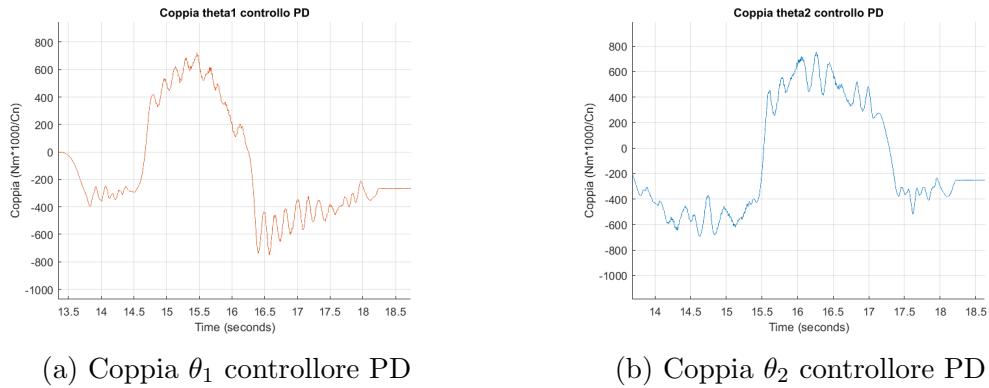


Figura 6.27: Coppie controllore PD braccia

con quelle coppie rispetto al riferimento, e di conseguenza possiamo guardare anche l'errore: Osservando i grafici si può notare che la posizione segue bene il setpoint

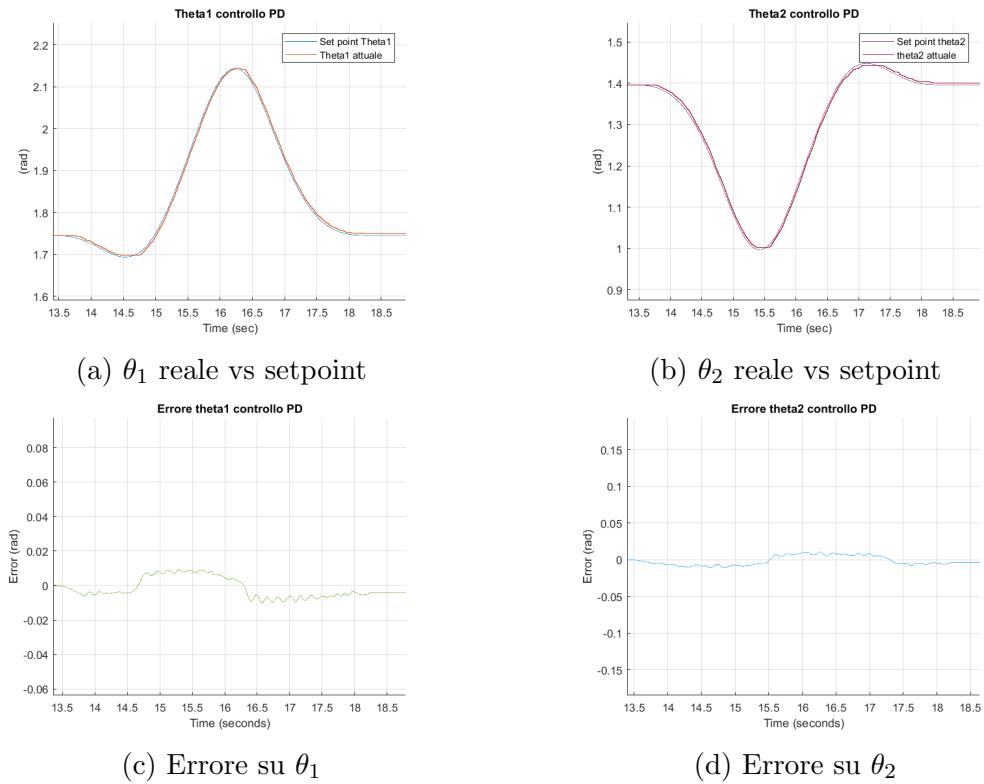


Figura 6.28: Andamenti posizione ed errori controllore PD

andiamo poi a rappresentare la traiettoria reale e il riferimento anche sull'asse [X,Y] ottenendo: [AGGIUNGI]

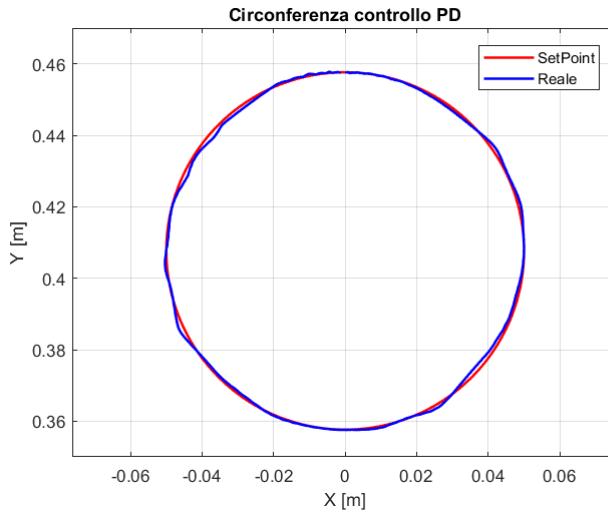


Figura 6.29: Cerchio assi [x,y] controllore PD

### 6.5.2 Controllo feed-forward con coppia pre-computata

Il passo successivo è stato quello di studiare ed implementare un controllore che opera in anello aperto; in particolare questo controllore calcola le coppie di disturbo basate sul modello matematico del sistema con parametri d'ingresso il *set-point* in posizione e velocità e accelerazione. L'introduzione di questi termini riesce a risolvere in maniera corretta il problema del tracciamento della traiettoria desiderata, di conseguenza gli elementi introdotti riescono a compensare gli effetti di accoppiamento presenti nel modello dinamico del sistema.

$$g_d = \Delta M(q_m^0) \ddot{q}_m^0 + C(q_m^0, \dot{q}_m^0) \dot{q}_m^0 \quad (6.10)$$

L'idea è quindi che i termini vengono calcolati considerando i valori di posizione, velocità ed accelerazione del riferimento, in quanto mediante una legge di moto sono sempre ben noti. L'elemento  $g_d$  compensa i termini di accoppiamento non lineari dovuti a forza d'inerzia, di Coriolis e centrifughe che dipendono dalla struttura e di conseguenza variano durante il movimento del manipolatore; in generale però calcolare questo termine è molto impegnativo, di conseguenza l'utilizzo di questo approccio su un sistema online può richiedere molto tempo, per questo solitamente vengono compensati solo i termini più importanti come quelli inerziali. A livello

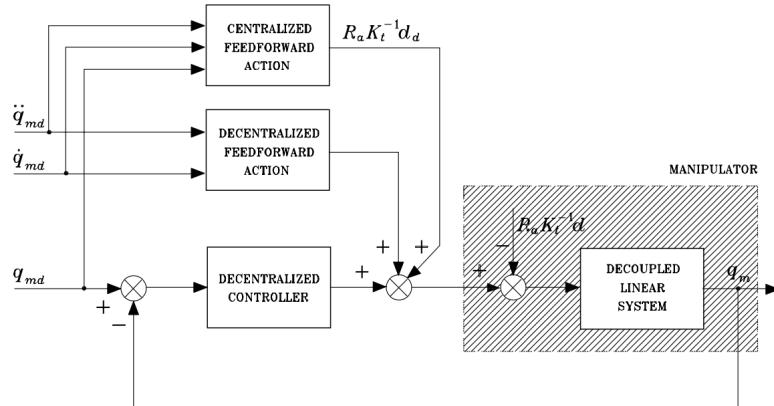


Figura 6.30: Schema teorico controllo feedforward

implementativo possiamo vederlo come:

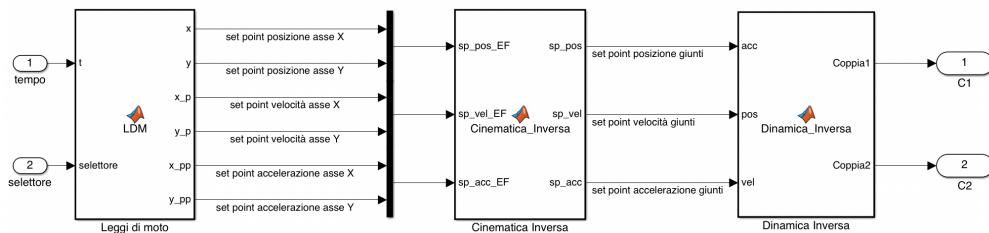


Figura 6.31: Controllore feedforward

A partire dalla legge di moto selezionata si ottengono i *set-point* all'end-effector, che grazie alla cinematica inversa vengono convertiti in riferimento ai giunti. Una volta ottenuti i riferimenti di posizione, velocità ed accelerazione si va ad applicare la funzione di dinamica inversa per calcolare le coppie di cui abbiamo bisogno. Le coppie sono quindi calcolate esclusivamente mediante dai riferimenti, è possibile vedere questo controllore come una versione di quello a dinamica inversa in anello aperto.

### 6.5.3 Controllo in dinamica inversa

Riprendendo le tecniche di controllo centralizzato caratteristiche della letteratura l'approccio successivo è stato quello del controllo in dinamica inversa. Il primo step

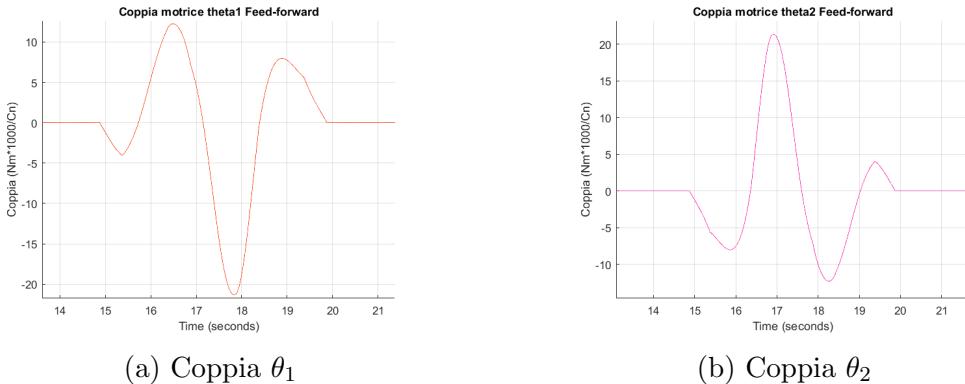


Figura 6.32: Coppie motori controllo fast-forward

è stato quello di prendere l'equazione della dinamica e ridefinirla come:

$$M(q_m)\ddot{q}_m + n(q_m, \dot{q}_m) = \tau_m \quad (6.11)$$

Dove  $n(q_m, \dot{q}_m)$  raccoglie i termini centrifughi e di Coriolis (in letteratura raccoglie anche i termini gravitazionali, ma come anticipato precedentemente nel nostro caso equivalgono a zero). L'idea del controllo a dinamica inversa si basa sul trovare il vettore di coppie  $\tau_m$  come funzione dello stato del sistema, cercando di creare una relazione ingresso-uscita di tipo lineare. Considerando che l'equazione della dinamica è lineare nel controllo e che la matrice d'inerzia è invertibile in ogni configurazione del manipolatore abbiamo la garanzia di trovare un controllore linearizzato di questo tipo. Possiamo andare a riscrivere il controllo come:

$$\tau_m = M(q_m)y + n(q_m, \dot{q}_m) \quad (6.12)$$

Dove  $y = \ddot{q}_m$  rappresenta un vettore d'ingresso con espressione ancora da determinare. La legge di controllo è basata sul calcolo della dinamica inversa del manipolatore, il sistema è lineare e disaccoppiato rispetto al nuovo ingresso, questo implica che le componenti  $y_k$  con  $k = -\infty \dots m-1$  influenzano solo la variabile  $q_m$  indipendentemente dal movimento degli altri giunti. In questo modo il problema del controllo è quello di trovare una legge  $\mathbf{y}$  stabilizzante. Viene quindi scelta:

$$y = K_p \tilde{q} + K_d \tilde{\dot{q}} + \ddot{q}_m^0 \quad (6.13)$$

Dove con i termini  $\tilde{q}$  e  $\tilde{\dot{q}}$  si indica la differenza tra il *setpoint* e la misurazione attuale di posizione e velocità.

$$\tilde{q} = q_m^0 - q_m \quad \tilde{\dot{q}} = \dot{q}_m^0 - \dot{q}_m \quad (6.14)$$

Possiamo andare ora a sostituire le definizioni di 6.14 in 6.13 e troviamo l'espressione:

$$\tilde{q} + K_d + \tilde{\dot{q}} = 0 \quad (6.15)$$

Chiaramente l'errore si verifica quando uno o entrambi i termini sono diversi da zero. Però, volendo assegnare la dinamica a ciascun giunto ci basta selezionare i guadagni delle matrici  $K_p$  e  $K_d$ , inoltre, se le due matrici sono definite positive è possibile ottenere un sistema asintoticamente stabile. Possiamo scegliere le matrici come:

$$K_p = \text{diag}(\omega_{01}^2, \dots, \omega_{0n}^2) \bar{M} \quad K_d = \text{diag}(2\xi_1\omega_{01}, \dots, 2\xi_n\omega_{0n}) \bar{M}$$

Grazie al controllo a dinamica inversa i termini di compensazioni vengono calcolati ad ogni iterazione, quindi a piccoli intervalli temporali, il loop interno serve per ottenere una relazione ingresso/uscita lineare e disaccoppiata mentre quello esterno grazie alla dinamica desiderata serve a stabilizzare il sistema. L'utilizzo di que-

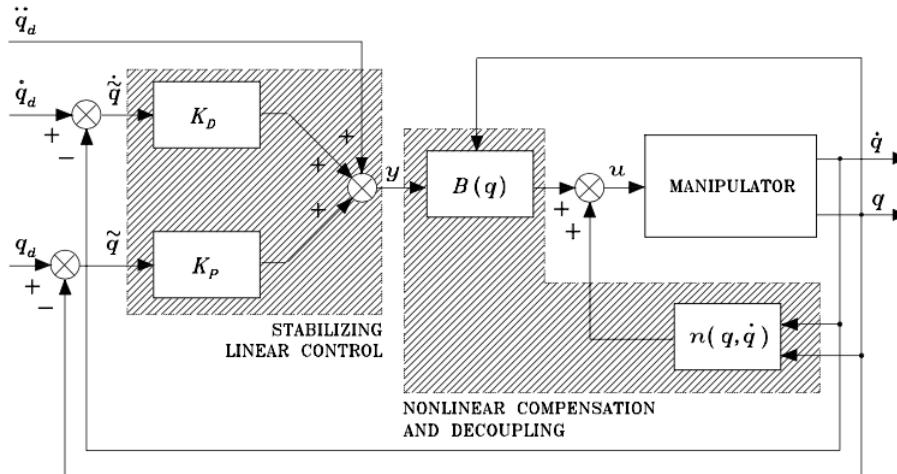


Figura 6.33: Schema controllore dinamica inversa

sto approccio è basato sull'ipotesi della cancellazione perfetta dei termini dinamici,

quindi, i parametri dinamici del sistema devono essere accuratamente conosciuti e l'equazione del moto deve essere calcolata in real-time. Possiamo andare a vedere lo schema implementato sul controllore come: Come per tutti i controllori visti fi-

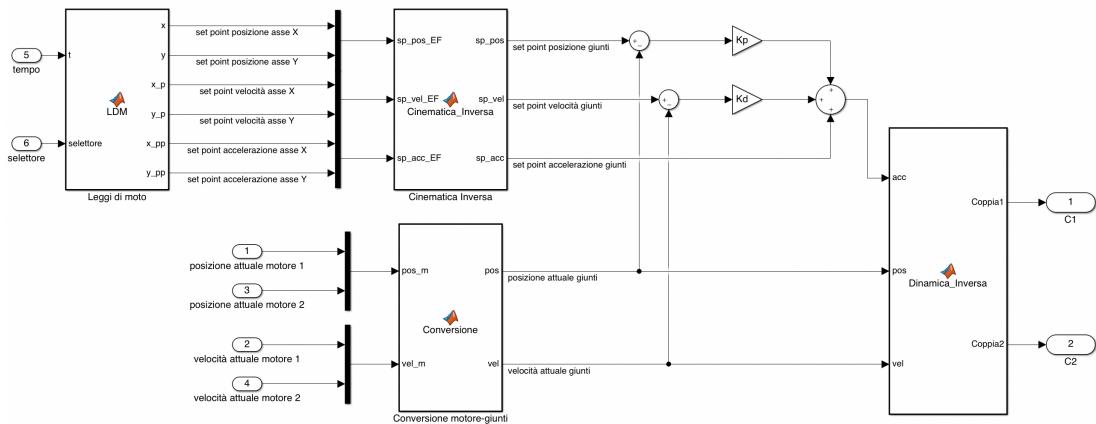
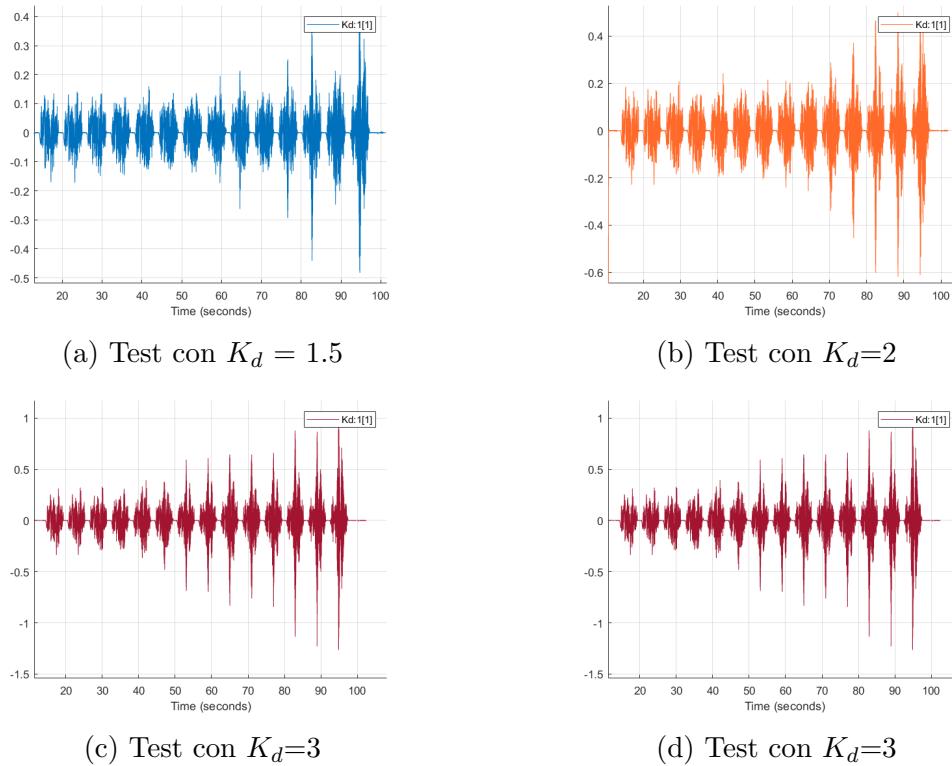


Figura 6.34: Controllore dinamica inversa

no ad ora a partire da selettore e dal tempo ed applicando la cinematica inversa andiamo ad ottenere i riferimenti di posizione, velocità ed accelerazione; andiamo poi a prendere le posizioni e le velocità attuali dei motori e le convertiamo da motori ai giunti. Successivamente andiamo per posizione e velocità a trovare gli errori che verranno moltiplicati rispettivamente per  $K_p$  e  $K_d$ . Per ottenere la  $y$  dobbiamo sommare questi due componenti insieme all'accelerazione del riferimento. Una volta ottenuto questo andiamo ad applicare la funzione di dinamica inversa con ingressi pari a  $ID(y, q, \dot{q})$ , da questa otteniamo la legge di controllo e le relative coppie che andranno assegnate ai link del manipolatore.

### Test $K_p$ e $K_d$ SISTEMARE

Per andare a scegliere  $K_p$  e  $K_d$  in modo preciso è stato adottato un approccio sperimentale, ovvero far variare  $K_p$ , in un range di valori compreso tra 350 e 950 ad intervalli di 50, in base ad un valore fisso di  $K_d$ , il tutto facendo fare una traiettoria circolare ed andando ad analizzare quando il controllo iniziava a dar origine a vibrazione. Andiamo ora a vedere un risultato di questo metodo.


 Figura 6.35: Test su  $K_d$ 

## Analisi

Anche in questo caso, come nei precedenti la traiettoria principale testata è stata quella del cerchio raggio 5cm in 5 secondi, possiamo andare a vedere le coppie fornite ai giunti: Come per il controllore PD possiamo andare a vedere e comparare la posizione attuale rispetto al riferimento e di conseguenza l'errore: Come per il controllore PD anche qua possiamo andare a rappresentare la traiettoria nel piano cartesiano, ottenendo: AGGIUNGI COMMENTI

### 6.5.4 Controllo robusto

L'effetto di incertezze sul modello induce in errore il sistema di controllo, soprattutto nel caso reale bisogna supporre che la compensazione del modello dinamico risulti imperfetta magari a causa di approssimazioni oppure per semplificazioni. Possiamo andare quindi a riscrivere l'equazione 6.11 come:

$$\tau_m = \hat{M}(q_m)y + \hat{n}(q_m, \dot{q}_m) \quad (6.16)$$

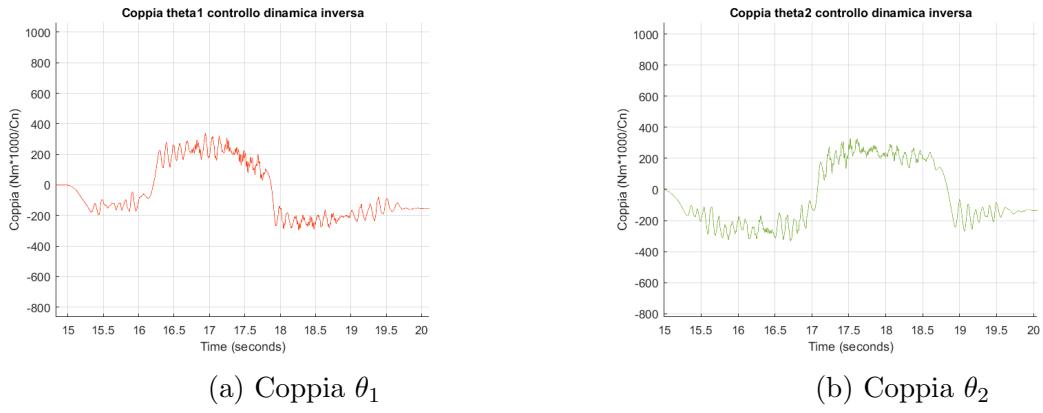


Figura 6.36: Coppie giunti controllo dinamica inversa

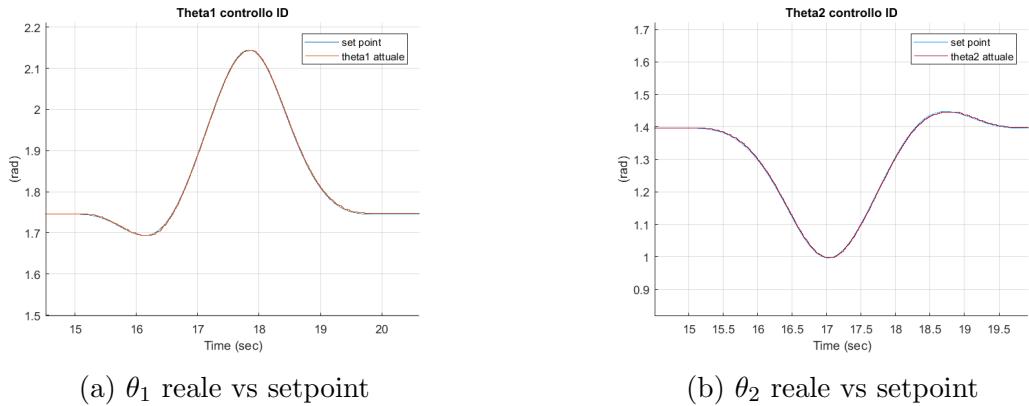


Figura 6.37: Posizione vs riferimento controllore ID

Dove  $\hat{M}$  e  $\hat{n}$  rappresentano i parametri stimati del modello dinamico, è possibile rappresentare l'incertezza come:

$$\bar{M} = \hat{M} - M \quad \bar{n} = \hat{n} - n$$

Riprendendo la legge di controllo vista prima possiamo riscriverla come:

$$M(q_m)\ddot{q}_m + n(q_m, \dot{q}_m) = \hat{M}(q_m)y + \hat{n}(q_m, \dot{q}_m) \quad (6.17)$$

Essendo la matrice  $M$  invertibile in ogni configurazione possiamo ricavare  $\ddot{q}_m$ :

$$\ddot{q}_m = y + (M^{-1}\hat{M} - I)y + M^{-1}\tilde{n} = y - \eta$$

## 6.5 Controllo braccia

---

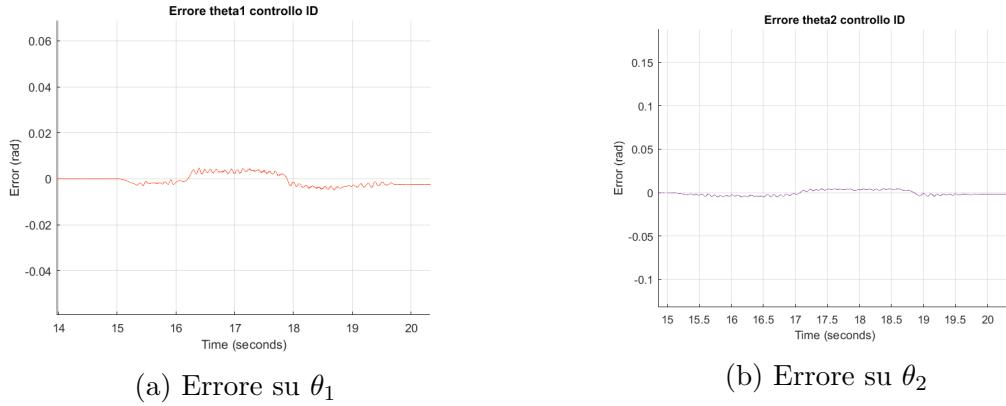


Figura 6.38: Errori controllore ID

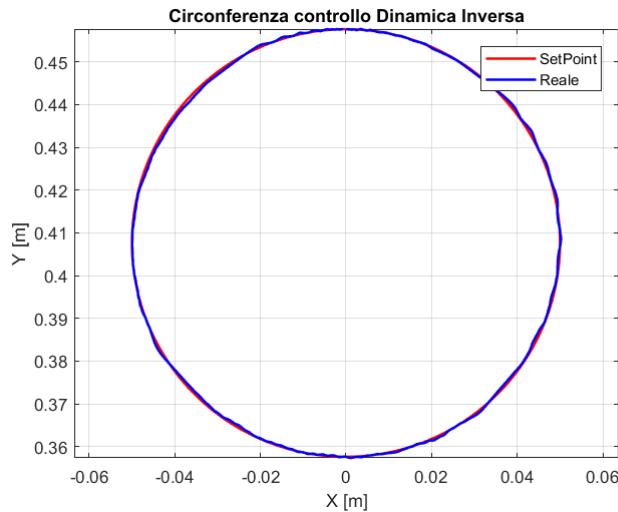


Figura 6.39: Cerchio  $[X, Y]$  controllo ID

Dove  $\eta$  è una funzione non lineare definita come:

$$\eta = (I - M^{-1}\hat{M})y - M^{-1}\tilde{n} \quad (6.18)$$

Adottando la legge 6.14 vista nel caso del controllore a dinamica inversa possiamo ottenere che l'errore dinamico è gestito dall'equazione:

$$\ddot{\tilde{q}} + K_d \dot{\tilde{q}} + K_p \tilde{q} = \eta \quad (6.19)$$

Otteniamo quindi un sistema non lineare e accoppiato, di conseguenza implementare un semplice controllore PD non basta; per risolvere questo problema occorre inserire

un termine non lineare, che sia funzione dell'errore e creato appositamente per fornire robustezza al controllo. Come per il controllore PD si ricerca una funzione grazie al metodo diretto di Lyapunov. Andiamo a definire lo stato del sistema come:

$$\xi = \begin{bmatrix} \tilde{q} \\ \dot{\tilde{q}} \end{bmatrix}$$

sostituendo poi lo stato nell'equazione  $\ddot{q}_m = y - \eta$  si ottiene un'equazione differenziale del primo ordine

$$\dot{\xi} = H\xi + D(\ddot{q}_m^0 - y + \eta) \quad (6.20)$$

Dove  $H$  e  $D$  sono definite come:  $H = \begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(2n \times 2n)}$ ,  $D = \begin{bmatrix} 0 \\ I \end{bmatrix} \in \mathbb{R}^{(2n \times n)}$

Si può vedere il problema di inseguimento della traiettoria come la soluzione che va a stabilizzare il sistema non lineare di qui sopra. Nella letteratura del controllore robusto, anche se l'incertezza  $\eta$  non è nota è comunque disponibile un suo intervallo di variazione. La legge  $\mathbf{y}$  dovrebbe garantire stabilità di  $\dot{\xi}$  per ogni  $\eta$  nell'intervallo. Di conseguenza vengono formulate tre assunzioni:

1.  $\sup_{t \geq 0} \|\ddot{q}_m^0\| < Q_M < \infty \forall \ddot{q}_m^0$
2.  $\|I - M^{-1}\hat{M}(q_m)\| \leq \alpha \leq 1 \forall q_m$
3.  $\|\tilde{n}\| \leq \Phi(\|\xi\|) < \infty \forall q_m, \dot{q}_m$

Andando ad analizzare le assunzioni possiamo dire che:

- La prima assunzione è soddisfatta sempre, in quanto per ogni traiettoria definita le accelerazioni non possono essere infinite
- La seconda assunzione conferma che  $M$  (e di conseguenza  $M^{-1}$ ) sia superiormente e inferiormente limitata, infatti

$$0 < M_m < \|M^{-1}(q_m)\| \leq M_M < \infty \forall q_m \quad (6.21)$$

Quindi, esiste sempre una scelta di  $\hat{M}$  che soddisfa la condizione, selezionando per esempio

$$\hat{M} = \frac{2}{M_M + M_m} I$$

otteniamo

$$||M^{-1}(q_m)\hat{M}(q_m) - I|| \leq \alpha = \frac{M_M - M_m}{M_M + M_m} < 1 \quad (6.22)$$

Il limite inferiore si ha quando  $\hat{M} = M$  in quanto  $\alpha = 0$ . Concentrandoci sull'assunzione 3, si può osservare che  $\tilde{n}$  è funzione di  $q_m, \dot{q}_m$ , nel primo caso, in base alla tipologia di giunto (rotoidale o prismatico) abbiamo intervalli che sono limitati e quindi il contributo è limitato. Anche la velocità è limitata grazie all'effetto della saturazione (esistente sulle velocità massime dei motori). Di conseguenza, prendiamo per  $\Phi$  una funzione calcolata nella norma dello stato come:

$$\Phi(||\xi||) = \alpha_0 + \alpha_1||\xi|| + \alpha_2||\xi||^2 \quad (6.23)$$

Andando a riprendere la legge di controllo 6.13 ed ampliandola possiamo definirla come:

$$y = \ddot{q}_m^0 + K_p \tilde{q} + K_d \dot{\tilde{q}} + \omega \quad (6.24)$$

Il termine PD ci assicura la stabilizzazione della dinamica della matrice dell'errore,  $\ddot{q}_m^0$  fornisce un termine di previsione e  $\omega$  è progettato in modo da combattere l'incertezza fornendo robustezza al sistema.

$$\dot{\xi} = \tilde{H}\xi + D(\eta - \omega) \quad (6.25)$$

Dove:

$$\tilde{H} = H - DK = \begin{bmatrix} 0 & I \\ -K_p & -K_d \end{bmatrix} \in \mathbb{R}^{(2n \times 2n)}$$

$$K_p = diag(\omega_{01}^2, \dots, \omega_{0n}^2) \quad K_d = diag(2\xi_1\omega_{01}^2, \dots, 2\xi_n\omega_{0n}^2)$$

Per andare a definire *omega* si procede col metodo diretto di Lyapunov; consideriamo come funzione:

$$V(\xi) = \xi^T Q \xi > 0, \forall \xi \neq 0 \quad (6.26)$$

con  $Q$  matrice simmetrica definita positiva. Facendo la derivata lungo la traiettoria si trova che

$$\dot{V} = \xi^T P \xi + 2z^T(\eta - \omega) \quad (6.27)$$

Considerando che  $\tilde{H}$  ha tutti gli autovalori con parte reale negativa, possiamo scegliere una qualunque  $P$  definita positiva che soddisfi:

$$\tilde{H}^T Q + Q \tilde{H} = -P$$

Consideriamo ora l'equazione 6.27 ,  $z = D^T Q \xi$ , il primo termine sulla parte di destra è definito negativo, di conseguenza la soluzione converge solo se  $\xi \in \mathbb{N}(D^T Q)$ , se invece non appartiene, il controllo  $\omega$  andrà scelto per rendere il secondo termine minore o uguale a zero utilizzando la legge di controllo:

$$\omega = \rho(|\xi|) \frac{z}{\|z\|}, \rho > 0 \quad (6.28)$$

con  $\rho$  funzione positiva da determinare. Scegliendo  $\omega$  in questo modo otteniamo:

$$z^T(\eta - \omega) = z^T \eta - \rho(|\xi|) \frac{z z^T}{\|z\|} \leq \|z\| [|\eta| \cdot \rho(|\xi|)]$$

se poi garantiamo che:  $|\xi| > |\eta|$  per ogni valore di posizione, velocità e velocità di riferimento, otteniamo che questo termine e  $\dot{V}$  sono negativi lungo tutte le traiettorie dell'errore; per poter soddisfare la diseguaglianza, dalla definizione di  $\eta$  troviamo:

$$\eta = (I - M^{-1} \hat{M})y - M^{-1} \tilde{n} \text{ e } y = \ddot{q}_m^0 + K\xi + \omega \quad (6.29)$$

sapendo poi che  $|\omega| = \rho$  troviamo che:

$$|\eta| \leq \|I - M^{-1} \hat{M}\| (\|\ddot{q}_m^0\| + \|K\| |\xi| + |\omega|) + \|M^{-1}\| \|\tilde{n}\|$$

Andando a maggiorare questa quantità troviamo che:

$$|\eta| \leq \alpha Q_M + \alpha \|K\| |\xi| + \alpha \rho(|\xi|) + M_M \Phi(|\xi|) < \rho(|\xi|) \quad (6.30)$$

è quindi possibile andare a selezionare un  $\rho$  in modo tale che:

$$\rho(||\xi||) \geq \frac{1}{1-\alpha} [\alpha Q_M + \alpha ||K|| ||\xi|| + M_M \Phi(||\xi||)] \quad (6.31)$$

Si può osservare che  $\Phi(||\xi||) = \alpha_0 + \alpha_1 ||\xi|| + \alpha_2 ||\xi||^2$  per soddisfare 6.31 basta che si selezionino:

$$\rho(||\xi||) = \beta_0 + \beta_1 ||\xi|| + \beta_2 ||\xi||^2$$

Con i valori:

$$\begin{cases} \beta_0 \geq \frac{\alpha Q_M + \alpha_0 M_M}{1-\alpha} \\ \beta_1 \geq \frac{\alpha K + \alpha_1 M_M}{1-\alpha} \\ \beta_2 \geq \frac{\alpha_2 M_M}{1-\alpha} \end{cases}$$

Andando poi a sostituire tutto nell'equazione 6.27 troviamo:

$$\dot{V} = -\xi^T P \xi + 2z^T (\eta - \rho(||\xi||) \frac{z}{||z||}) < 0 \quad \forall \xi \neq 0 \quad (6.32)$$

Di conseguenza  $\xi = 0$  è un equilibrio di stato asintoticamente e globalmente stabile. La legge di controllo è quindi formata da tre termini, un primo termine che compensa i termini non lineari e gli accoppiamenti tra i link, un secondo termine che stabilizza il sistema dinamico grazie ad una retroazione con previsione per l'errore e un terzo termine che fornisce robustezza contrastando le incertezze e calcolando i termini non lineari che dipendono dallo stato del manipolatore.

Ad alte frequenze però potrebbe esserci la commutazione della variabile di controllo, che potrebbe portare ad un'elevata presenza di "sali/scendi" a causa del terzo termine, per questo possiamo approssimarla come:

$$\omega = \begin{cases} \rho(||\xi||) \frac{z}{||z||} & \text{for } ||z|| \geq \varepsilon \\ \rho(||\xi||) \frac{z}{\varepsilon} & \text{for } ||z|| < \varepsilon \end{cases} \quad (6.33)$$

Possiamo andare quindi a rappresentare lo schema del controllo robusto teorico nel seguente modo: Data la difficoltà implementativa del controllore robusto sul modello reale sono state condotte due tipologie di analisi: la prima è stata un'analisi del

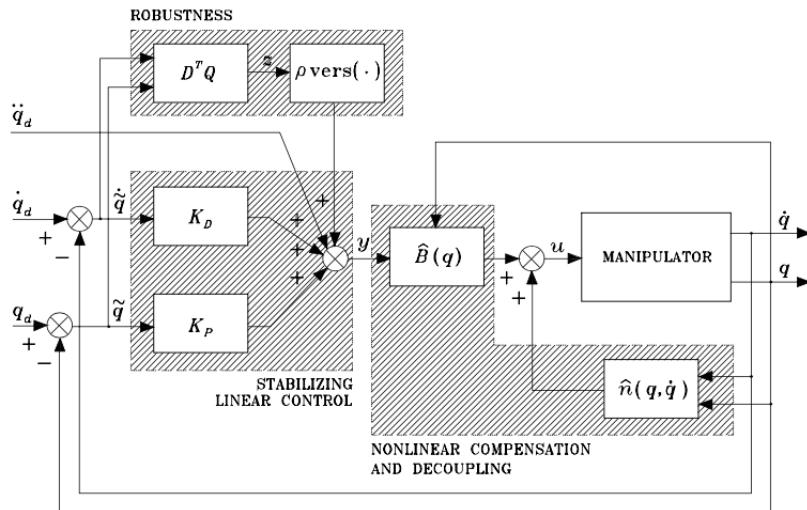


Figura 6.40: Schema controllore robusto

comportamento del controllore grazie alla modellazione di un sistema più semplice, in particolare il sistema ad un grado di libertà del manovellismo; la seconda invece è stata la simulazione del controllore sul modello simulink del manipolatore.

### Analisi manovellismo

Come è stato anticipato precedentemente è stata fatta un'analisi mediante il sistema ad un grado di libertà biella manovella, [PRESENTA RISULTATI]

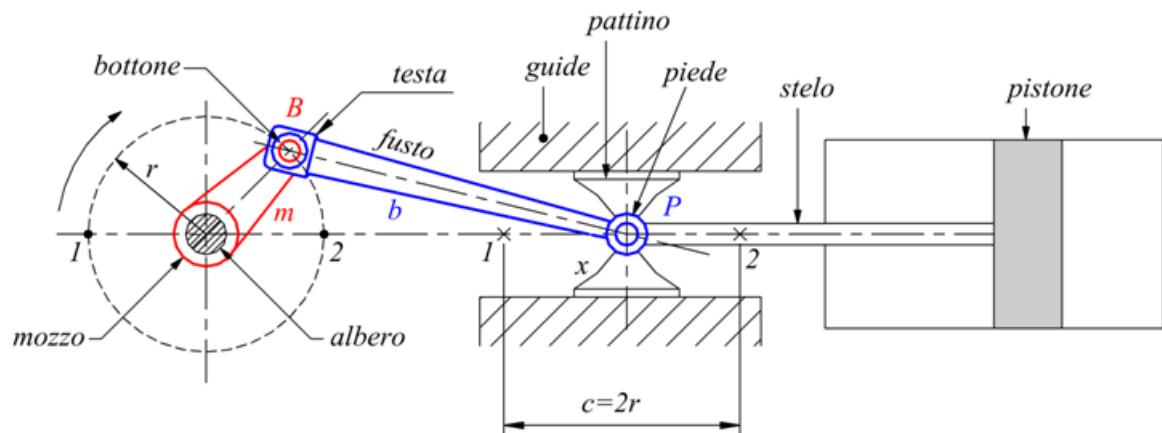


Figura 6.41: Sistema biella manovella

## 6.6 Confronto approcci controllori

Avendo introdotto e verificato le principali tipologie di controllo centralizzato è venuto il momento di fare un confronto andando a vedere quale tra gli approcci visti è il migliore. In particolare, sulla traiettoria effettuata andiamo a confrontare gli errori tra il controllore PD e quello ID: Da entrambe le immagini è chiaro come è il

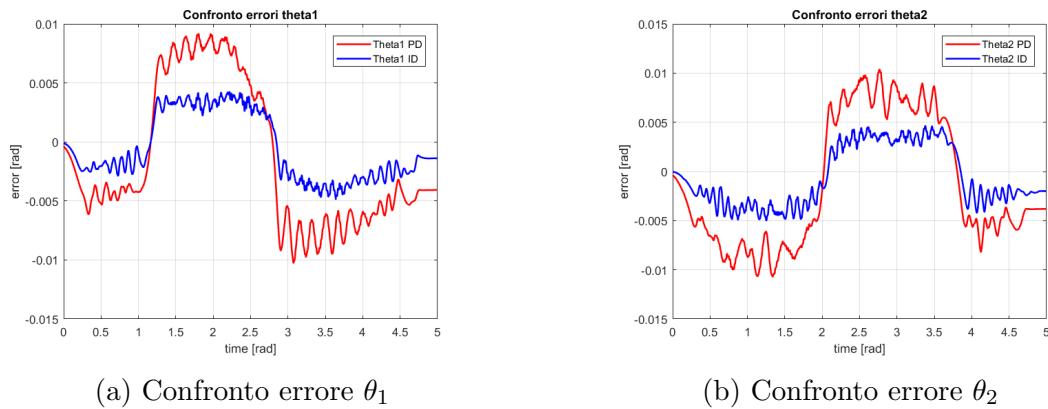


Figura 6.42: Confronto controllore PD e ID

controllore a dinamica inversa a prevalere. Di conseguenza si è scelto questo come controllore definitivo e sono stati analizzati altri andamenti con leggi di moto diverse per valutare il comportamento del controllore.

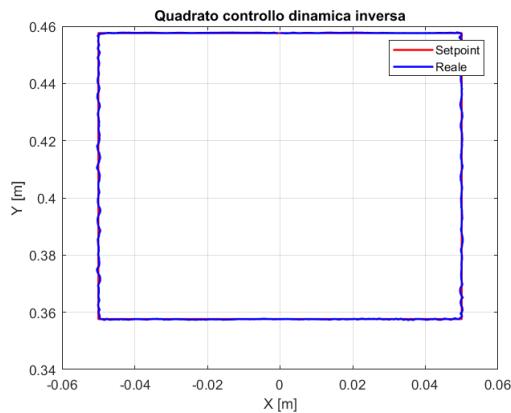


Figura 6.43: Quadrato controllo dinamica inversa

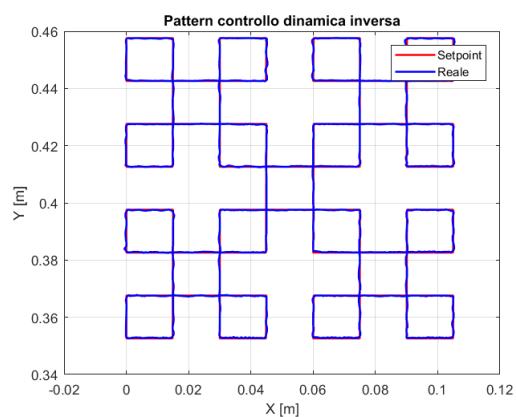


Figura 6.44: Pattern controllo dinamica inversa

---

## **7 Conclusioni**

### **7.1 Confronto modellazione teorica-pratica**

### **7.2 Sviluppi futuri**

## Elenco delle figure

1.1	Robot PKM	6
2.1	Rappresentazione fisica del robot	9
2.2	Equazioni alle circonferenze	10
3.1	Confronto dinamica leggi di moto su accelerazioni	22
3.2	Confronto dinamica leggi di moto su velocità	22
3.3	Confronto dinamica leggi di moto su posizioni	22
3.4	Modello simulink manipolatore	23
3.5	Modello Adams manipolatore 5R	23
3.6	Coppie in uscita Simulink	24
3.7	Coppie in uscita Adams	24
3.8	Errore Simulink-Adams	25
3.9	Caso 1 singolarità	26
3.10	Caso 2 singolarità	26
3.11	Caso 3 singolarità	27
3.12	Caso 4 e 5 singolarità	28
3.13	Punti di singolarità	29
3.14	Numero di condizionamento	30
3.15	Angoli di movimentazione dei giunti motorizzati	31
3.16	<i>Workspace</i> 5R	31
3.17	Numero di condizionamento workspace	32
4.1	Struttura vite	33
4.2	Utensile da disegno	33
4.3	Coppie e posizione end-effector	35
5.1	Schema azionamento	39
5.2	Registri azionamento	39
6.1	Banco di test	41
6.2	Schema modulo bechoff	43
6.3	Topologia della rete	43
6.4	Configurazione rete ethernet user PC	44

## ELENCO DELLE FIGURE

---

6.5	Topologia rete mediante Ec-engineer . . . . .	44
6.6	PDO in input . . . . .	45
6.7	PDO Output 1 e 2 . . . . .	45
6.8	Schema generale simulink . . . . .	46
6.9	Fase 1: Inizializzazione . . . . .	46
6.10	Fase 2: Input . . . . .	47
6.11	Conversione e lettura motori . . . . .	48
6.12	Fase 3: Stateflow . . . . .	48
6.13	Fase 4: Schema di controllo della vite . . . . .	49
6.14	Fase 5: Schema di controllo delle braccia . . . . .	50
6.15	Schema espanso invio coppia . . . . .	51
6.16	Fase 6: Copie in uscita . . . . .	51
6.17	Fase di Homing . . . . .	53
6.18	Fase di posizionamento . . . . .	55
6.19	Implementazione funzione di posizionamento . . . . .	56
6.20	Fase di controllo . . . . .	57
6.21	Interfaccia grafica . . . . .	59
6.22	Schema teorico controllore proporzionale . . . . .	61
6.23	CAMBIARE . . . . .	61
6.24	Schema teorico controllore PD . . . . .	62
6.25	Controllore PD Vite . . . . .	62
6.26	Controllore PD braccia . . . . .	65
6.27	Copie controllore PD braccia . . . . .	66
6.28	Andamenti posizione ed errori controllore PD . . . . .	66
6.29	Cerchio assi [x,y] controllore PD . . . . .	67
6.30	Schema teorico controllo feedforward . . . . .	68
6.31	Controllore feedforward . . . . .	68
6.32	Copie motori controllo fast-forward . . . . .	69
6.33	Schema controllore dinamica inversa . . . . .	70
6.34	Controllore dinamica inversa . . . . .	71
6.35	Test su $K_d$ . . . . .	72

6.36	Coppie giunti controllo dinamica inversa . . . . .	73
6.37	Posizione vs riferimento controllore ID . . . . .	73
6.38	Errori controllore ID . . . . .	74
6.39	Cerchio [X,Y] controllo ID . . . . .	74
6.40	Schema controllore robusto . . . . .	79
6.41	Sistema biella manovella . . . . .	79
6.42	Confronto controllore PD e ID . . . . .	80
6.43	Quadrato controllo dinamica inversa . . . . .	80
6.44	Pattern controllo dinamica inversa . . . . .	81

## Elenco delle tabelle

1	Parametri manipolatore . . . . .	9
2	Leggi di moto validazione . . . . .	24
3	Parametri end-effector . . . . .	34
4	Tipologie controllo azionamenti . . . . .	47
5	Valori variabile <i>bool</i> . . . . .	58
6	Valori luci . . . . .	58