

POLITECNICO DI TORINO

A.A. 2015/2016



DISPOSITIVI E SISTEMI ROBOTICI

ESERCITAZIONE:

ROBOT ABB MODELLO IRB 120

DOCENTE:

PROF. STEFANO PAOLO PASTORELLI

STUDENTE:

GUGLIELMO CERVETTINI



## Sommario

1. INTRODUZIONE	4
2. SISTEMI DI RIFERIMENTO E PARAMETRI DI DENAVIT-HARTENBERG	6
3. TRAIETTORIA	9
3.1 TRAIETTORIA GRADI DI LIBERTA' NEI GIUNTI	9
3.2 TRAIETTORIA E PERCORSO NELLO SPAZIO OPERATIVO DEL CENTRO POLSO CP, DELL'END EFFECTOR EE E DEL TOOL CENTRE POINT TCP	10
4. TENSORI DI INERZIA	16
5. AZIONI MOTRICI	20
6. SPAZIO DI LAVORO	24
7. ALGORITMO DI CINEMATICA INVERSA	26
8. PIANIFICAZIONE TRAIETTORIA RETTILINEA TOOL CENTRE POINT TCP	29
9. ANALISI CONFIGURAZIONI DI SINGOLARITA'	35
ALLEGATO 1	40
ALLEGATO 2	42
ALLEGATO 3	57
ALLEGATO 4	59
ALLEGATO 5	61
ALLEGATO 6	62

## 1. INTRODUZIONE

Lo scopo del presente lavoro è quello di effettuare un'analisi cinematica e dinamica e lo studio di particolari traiettorie del robot **ABB MODELLO IRB 120**, illustrato nell'allegato 1.

E' un robot a 6 assi, con struttura a braccio articolato (3 gradi di libertà) e polso sferico (3 gradi di libertà).

In figura 1 è raffigurato il robot con indicazione degli assi di rotazione e dei rispettivi gradi di libertà nei giunti:  $q_1, q_2, q_3, q_4, q_5$ , e  $q_6$ ; il robot è rappresentato nella configurazione di *zero assi* (tutti i gradi di libertà nei giunti nulli).

In figura 2 sono indicate le quote caratteristiche di posizione degli assi giunti, oltre ad alcuni punti particolari:

- A = intersezione assi giunti 1 e 2
- B = intersezione assi giunti 1 e 3 (nella configurazione di *zero assi*)
- CP = intersezione assi giunti 4, 5 e 6 (centro polso)
- EE = punto *end effector* del polso coincidente con il centro della flangia di collegamento al tool.

Sulla base del robot è assunto il sistema di riferimento di base di assi  $X_0, Y_0, Z_0$  indicato in figura 2.

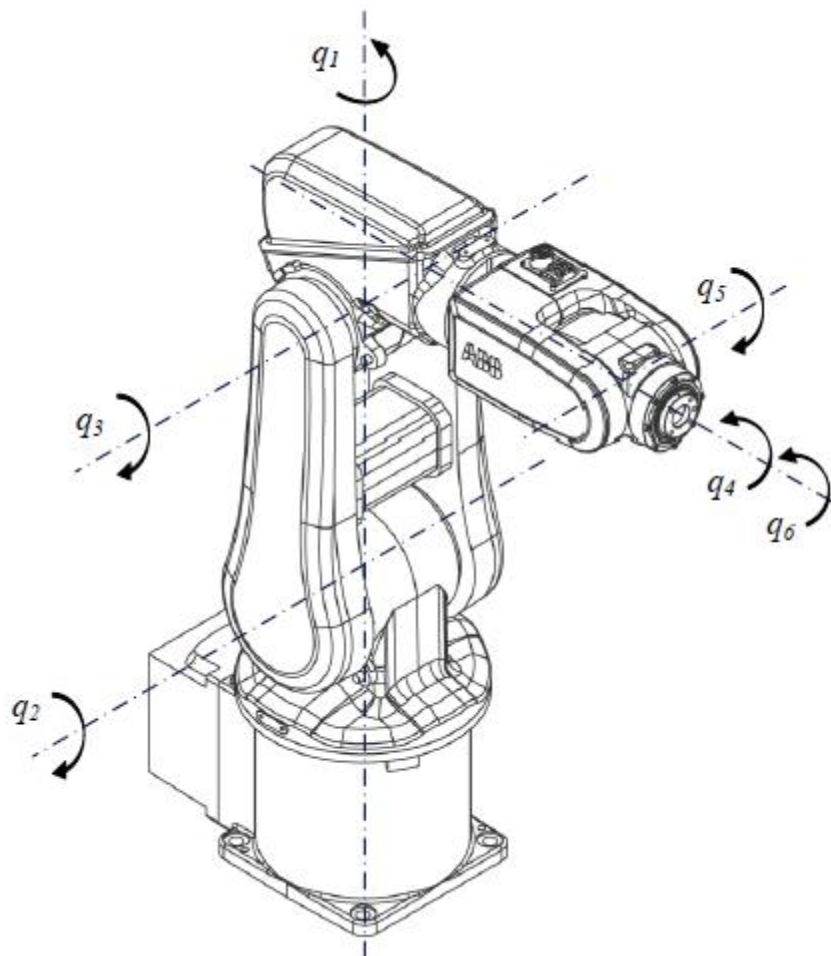


Figura 1: Assi di rotazione e rispettivi gradi di libertà

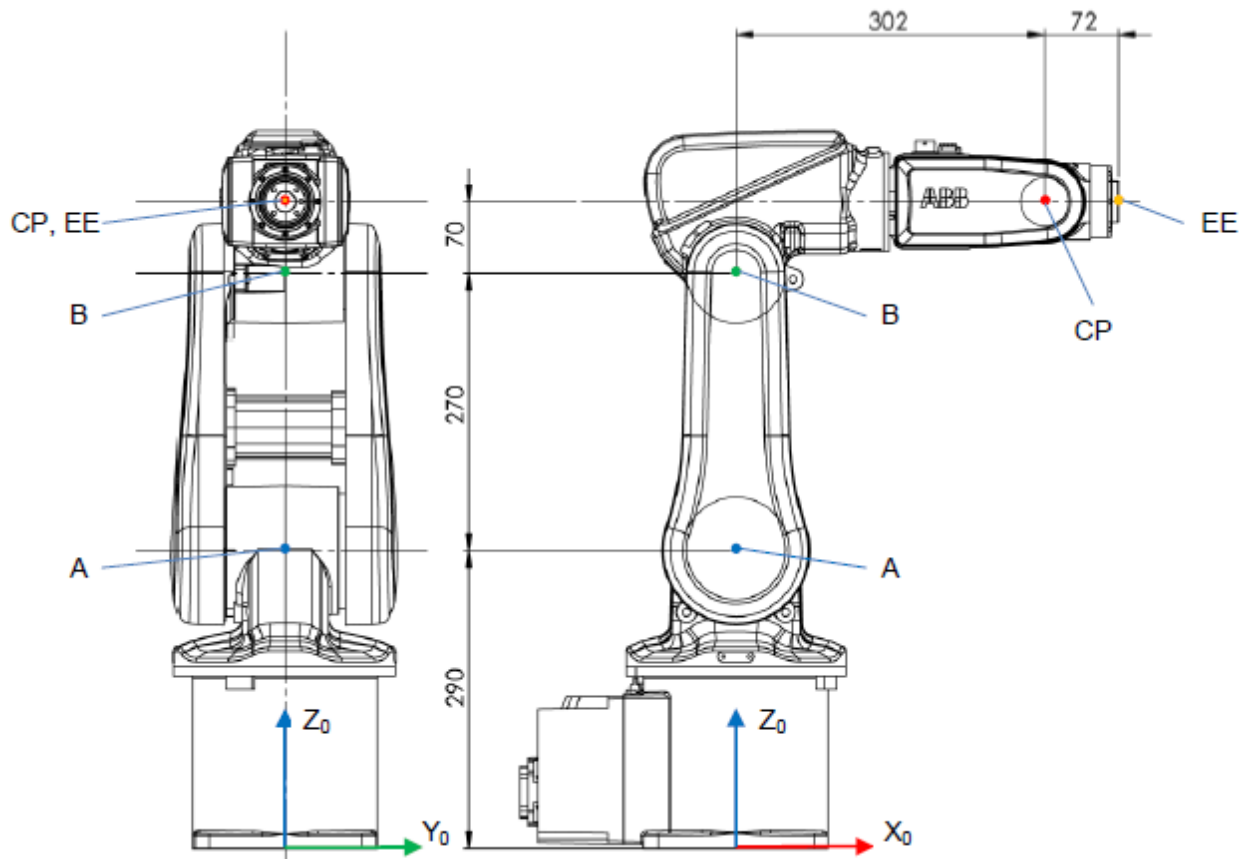


Figura 2: Quote e punti caratteristici

Si assume installato sulla flangia del polso un ugello di verniciatura, nella posizione e con le quote caratteristiche indicate in figura 3. L'asse  $Z_T$  rappresenta l'asse di verniciatura dell'ugello e T il punto di riferimento del tool.

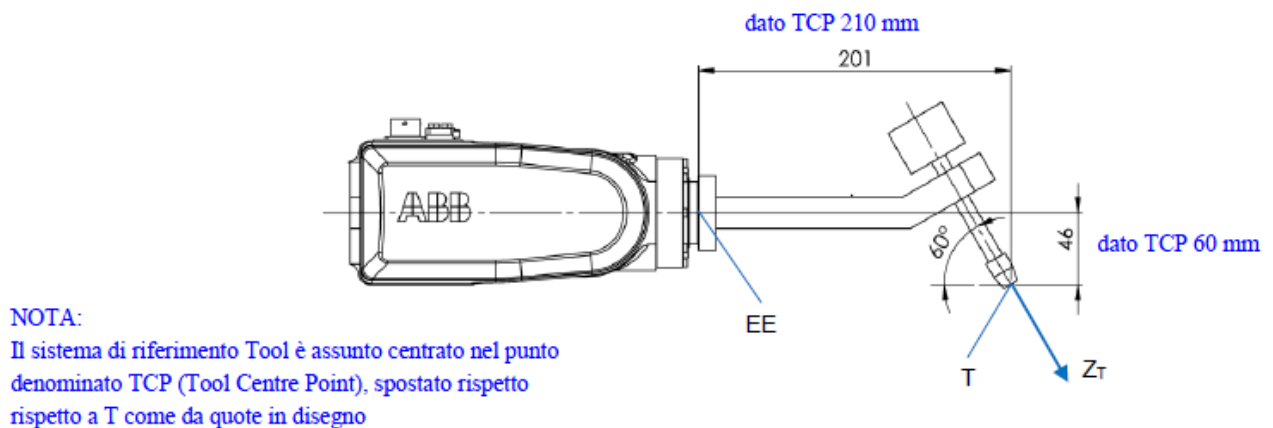


Figura 3: Tool ugello di verniciatura

## 2. SISTEMI DI RIFERIMENTO E PARAMETRI DI DENAVIT-HARTENBERG

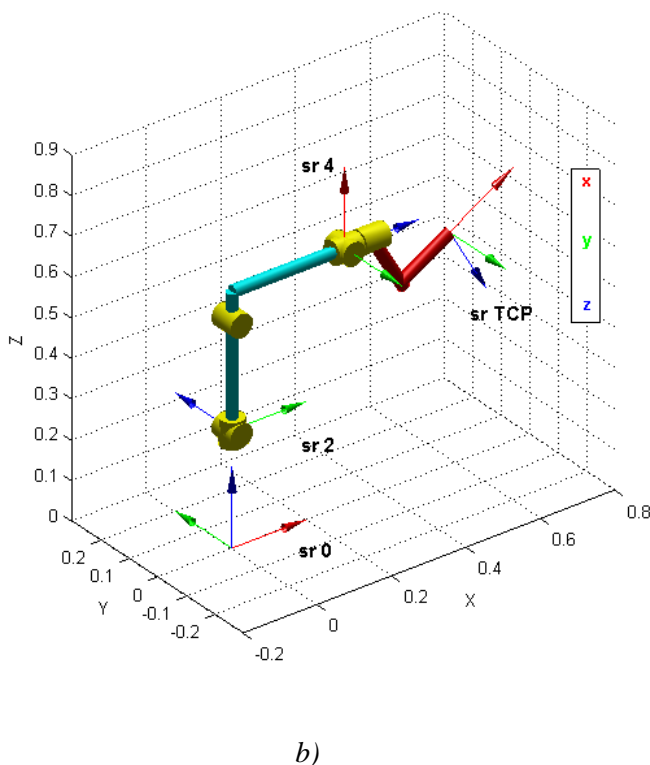
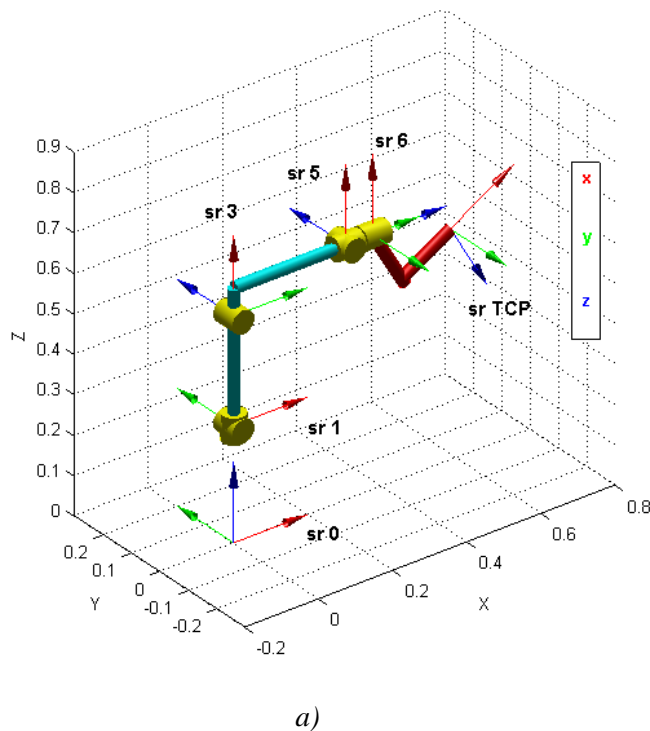


Figura 4: Rappresentazione wire-frame Robot

Nella figura 4 sono rappresentati i sistemi di riferimento scelti secondo D-H: nella figura a) si riportano quelli relativi ai link 0 (base), 1, 3, 5, 6 e TCP (*tool centre point*); nella figura b) si riportano quelli relativi ai link 2 e 4.

I sistemi di riferimento di ogni corpo della struttura multi-body sono stati definiti secondo la convenzione di Denavit-Hartenberg modificata (versione di John Craig).

Tale convenzione consente di definire la posa relativa di due sistemi di riferimento nello spazio attraverso l'utilizzo di soli quattro parametri geometrici, anziché sei come nel caso più generale; questo è possibile perché tra due sistemi di riferimento consecutivi è presente un giunto ad un solo grado di libertà.

I parametri di Denavit-Hartenberg sono:

- $\alpha_{i-1}$ : angolo di twist tra l'asse z di un corpo e quello del corpo successivo, positivo secondo l'asse x del corpo prossimale in base alla regola della mano destra;
- $a_{i-1}$ : distanza di offset tra l'asse z di un corpo e quello del corpo successivo, misurata lungo l'asse x del corpo prossimale;
- $d_i$ : distanza fra l'asse x di un corpo e quello del corpo successivo misurata lungo l'asse z del corpo distale;
- $\theta_i$ : angolo di rotazione fra l'asse x di un corpo e quello del corpo successivo misurato secondo la regola della mano destra rispetto l'asse z del corpo distale.

Nel seguito è riportata la tabella contenente i parametri di D-H:

$i$	$\alpha_{i-1} [deg]$	$a_{i-1} [mm]$	$d_i [mm]$	$\vartheta_i [deg]$	
1	0	0	$d_1 = 290$	$q_1$	${}^0\hat{A}_1$
2	$\alpha_1 = -90$	0	0	$q_2 - 90$	${}^1\hat{A}_2$
3	0	$a_2 = 270$	0	$q_3$	${}^2\hat{A}_3$
4	$\alpha_3 = -90$	$a_3 = 70$	$d_4 = 302$	$q_4$	${}^3\hat{A}_4$
5	$\alpha_4 = 90$	0	0	$q_5$	${}^4\hat{A}_5$
6	$\alpha_5 = -90$	0	$d_6 = 72$	$q_6$	${}^5\hat{A}_6$

Figura 5: Parametri di Denavit-Hartenberg

$${}^6\hat{A}_{TCP} = \begin{bmatrix} c_{\alpha_{TCP}} & 0 & -s_{\alpha_{TCP}} & -h \\ 0 & 1 & 0 & 0 \\ s_{\alpha_{TCP}} & 0 & c_{\alpha_{TCP}} & f \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{aligned} \alpha_{TCP} &= 60^\circ \\ f &= 210 \text{ mm} \\ h &= 60 \text{ mm} \end{aligned}$$

Figura 6: Parametri geometrici tool e sua matrice di posa rispetto al corpo precedente

Definiti tali parametri, è stato possibile procedere alla schematizzazione del robot in ambiente MATLAB tramite la funzione “SerialLink” del *robotics toolbox* di Peter Corke. Si riporta un estratto del listato compilato (allegato 2):

```
%% LIMITI DEI GRADI DI LIBERTA' DEI GIUNTI (DA CATALOGO)

qlim1=[-165 165]*pi/180;
qlim2=[-110 110]*pi/180;
qlim3=[-110 70]*pi/180;
qlim4=[-160 160]*pi/180;
qlim5=[-120 120]*pi/180;
qlim6=[-400 400]*pi/180;

%% LINK DEL ROBOT

L(1)=Link('alpha', alfa1, 'a', a1, 'd', d1, 'offset', teta10, 'modified', 'qlim', qlim1);
L(2)=Link('alpha', alfa2, 'a', a2, 'd', d2, 'offset', teta20, 'modified', 'qlim', qlim2);
L(3)=Link('alpha', alfa3, 'a', a3, 'd', d3, 'offset', teta30, 'modified', 'qlim', qlim3);
L(4)=Link('alpha', alfa4, 'a', a4, 'd', d4, 'offset', teta40, 'modified', 'qlim', qlim4);
L(5)=Link('alpha', alfa5, 'a', a5, 'd', d5, 'offset', teta50, 'modified', 'qlim', qlim5);
L(6)=Link('alpha', alfa6, 'a', a6, 'd', d6, 'offset', teta60, 'modified', 'qlim', qlim6);

ABBirb120=SerialLink(L, 'name', 'ABBirb120');
```

E' bene notare che, per i parametri di twist e offset, si è usata nel listato una notazione differente rispetto a quella riportata in tabella (figura 5). Inoltre, i limiti di ogni grado di libertà nei giunti sono stati rilevati dal catalogo del robot (allegato 1).



Per ottenere una visualizzazione più realistica del robot sono stati utilizzati file CAD dei vari link.

Ognuno di questi corpi ha un sistema di riferimento ereditato dal CAD.

In generale si è dunque reso necessario ricavare per ogni link, l'informazione di posa relativa fra sistema di riferimento CAD e di D-H (rappresentata dalla matrice costante  ${}^{DHi}\hat{A}_{CADi}$ , ricavata da considerazioni grafiche).

$${}^0\hat{A}_{CADi} = {}^0\hat{A}_{DHi} * {}^{DHi}\hat{A}_{CADi} \quad (1)$$

Successivamente, attraverso la funzione “*fkine*” (metodo di “*SerialLink*”), è stata calcolata la posa dei sistemi di D-H di ogni corpo rispetto al sistema di riferimento zero fisso ( ${}^0\hat{A}_{DHi}$ , variabile a seconda della configurazione).

Figura 7: Schematizzazione CAD  
Mechanics Explorer

Infine, utilizzando l'equazione 1, sono state ricavate le matrici dei sistemi CAD rispetto al sistema zero fisso, le quali consentono la visualizzazione finale del robot.

In figura 7 è riportata la configurazione con un certo vettore di gradi di libertà:

$$\underline{q} = [20, 30, 10, -30, 30, -20].$$



### 3. TRAIETTORIA

Si desidera effettuare un movimento lungo una traiettoria definita con leggi temporali dei gradi di libertà nei giunti come indicato nel file *traiettoria\_6gdl\_v01.mat* contenente i seguenti vettori (essendo  $\# = 1, 2, \dots, 6$  il numero identificativo del giunto):

**tempovet** = vettore tempo [s]

**q#vet** = vettore di posizione angolare del grado di libertà # [deg]

**q#pvet** = vettore di velocità angolare del grado di libertà # [deg/s]

**q#ppvet** = vettore di accelerazione angolare del grado di libertà # [deg/s<sup>2</sup>]

#### 3.1 TRAIETTORIA GRADI DI LIBERTA' NEI GIUNTI

Nelle seguenti figure vengono riportati gli andamenti temporali dei gradi di libertà nei giunti e le relative velocità e accelerazioni durante il movimento secondo la traiettoria assegnata.

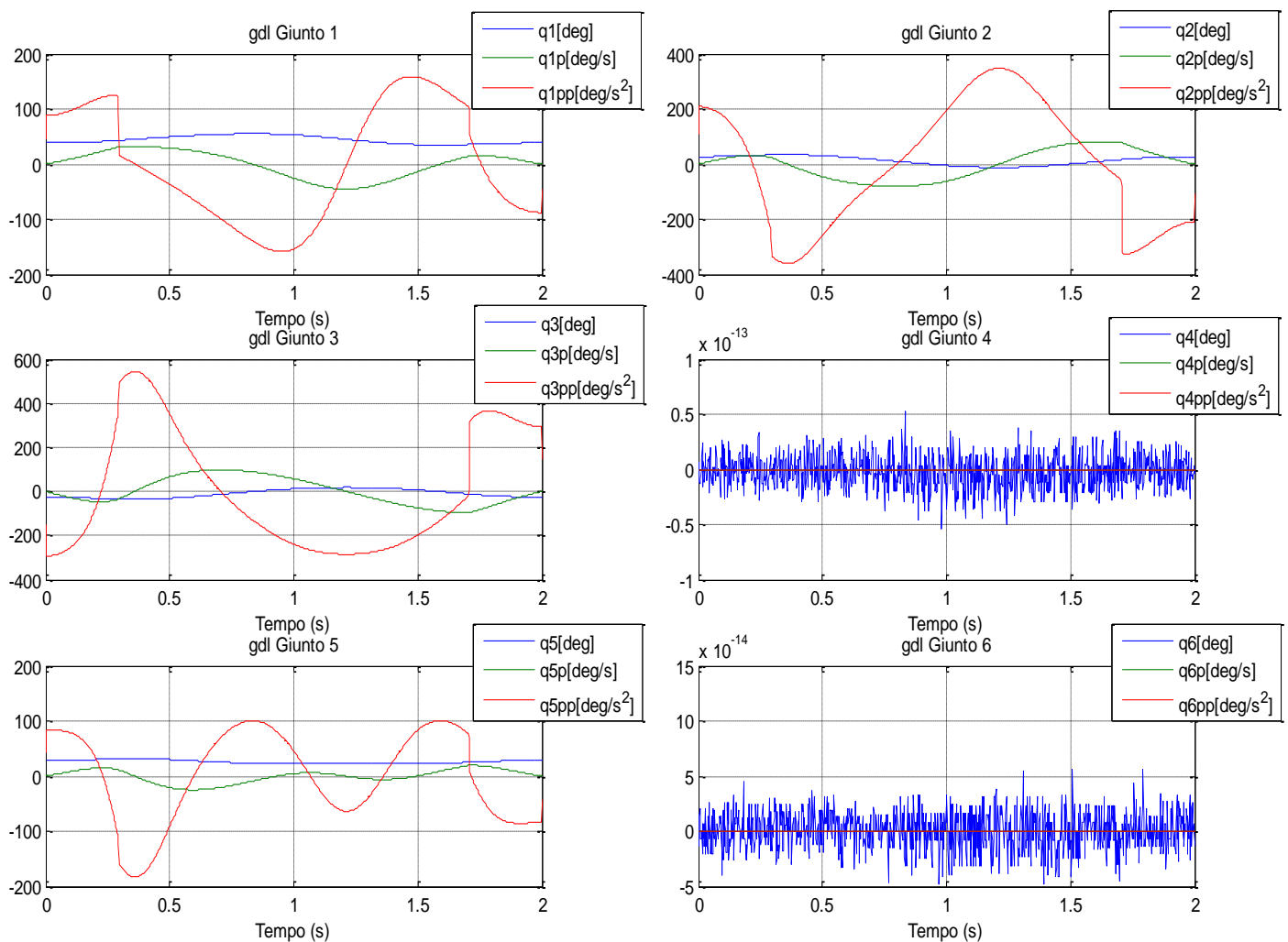
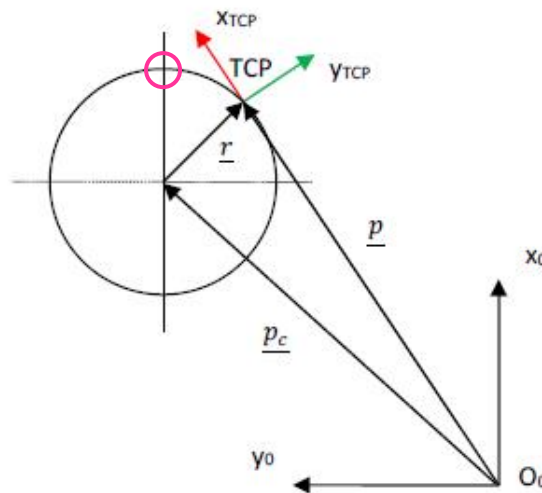


Figura 8: Traiettoria gradi di libertà nei 6 giunti

### 3.2 TRIAETTORIA E PERCORSO NELLO SPAZIO OPERATIVO DEL CENTRO POLSO CP, DELL'END EFFECTOR EE E DEL TOOL CENTRE POINT TCP

Il percorso del *tool centre point* TCP che si vuole realizzare è di tipo circolare su piano parallelo al piano  $\langle x_0 y_0 \rangle$  (sistema di riferimento zero fisso), con asse  $z_{TPC}$  orientato in verso opposto a  $z_0$  ed asse  $x_{TCP}$  orientato nella direzione di  $\overrightarrow{O_0 TCP}$  proiettato sul piano del moto:



Espressione del percorso di posizione di TCP:

$$\underline{p}(s) = \underline{p}_c + \underline{r}(s) = \underline{p}_c + r \begin{bmatrix} \cos(2\pi s) \\ \sin(2\pi s) \\ 0 \end{bmatrix} \quad \text{parametro di curva } s: 0 \leq s \leq 1$$

Espressione della traiettoria di posizione di TCP:

$$\underline{p}(t) = \underline{p}_c + r \begin{bmatrix} \cos(2\pi s(t)) \\ \sin(2\pi s(t)) \\ 0 \end{bmatrix} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} \quad |p_{xy}| = \sqrt{p_x^2 + p_y^2}$$

Espressione della traiettoria di posizione ed orientazione del S.R. Tool/TCP in forma omogenea:

$${}^0\hat{A}_{TCP}(t) = \begin{bmatrix} \frac{p_x}{|p_{xy}|} & \frac{p_y}{|p_{xy}|} & 0 & p_x \\ \frac{p_y}{|p_{xy}|} & -\frac{p_x}{|p_{xy}|} & 0 & p_y \\ 0 & 0 & -1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Espressione della traiettoria di posizione ed orientazione del S.R. 6 in forma omogenea:

$${}^0\hat{A}_6(t) = {}^0\hat{A}_{TCP} {}^{TCP}\hat{A}_6 = {}^0\hat{A}_{TCP} {}^6\hat{A}_{TCP}^{-1}$$

Figura 9: Percorso tool centre point TCP

Per poter tracciare il percorso e la traiettoria dei componenti di interesse, è stata utilizzata la funzione “fkine” che riceve in ingresso i valori dei gradi di libertà nei giunti per ogni istante della traiettoria assegnata:

```
%% TRAIETTORIA CON PERCORSO CIRCOLARE

load('rvctools\traiettoria_6gdl_v02.mat');

for i=1:length(tempovet)

    qvec_i=deg2rad([q1vet(i),q2vet(i),q3vet(i),q4vet(i),q5vet(i),q6vet(i)]);

    [zeroAtcp, A_all]=ABBirb120.fkine(qvec_i);

    zeroCP(:,i)=A_all(1:3,4,4);
    zeroEE(:,i)=A_all(1:3,4,6);
    zeroTcp(:,i)=zeroAtcp(1:3,4);

end
```

In questo modo, viene calcolato, istante per istante, il vettore posizione del centro polso, dell’*end effector* e del *tool centre point* rispetto al sistema di riferimento zero (fisso).

Nelle figure 10, 11 e 12 sono riportate la traiettoria e il percorso del centro polso, dell’*end effector* e del *tool centre point*.

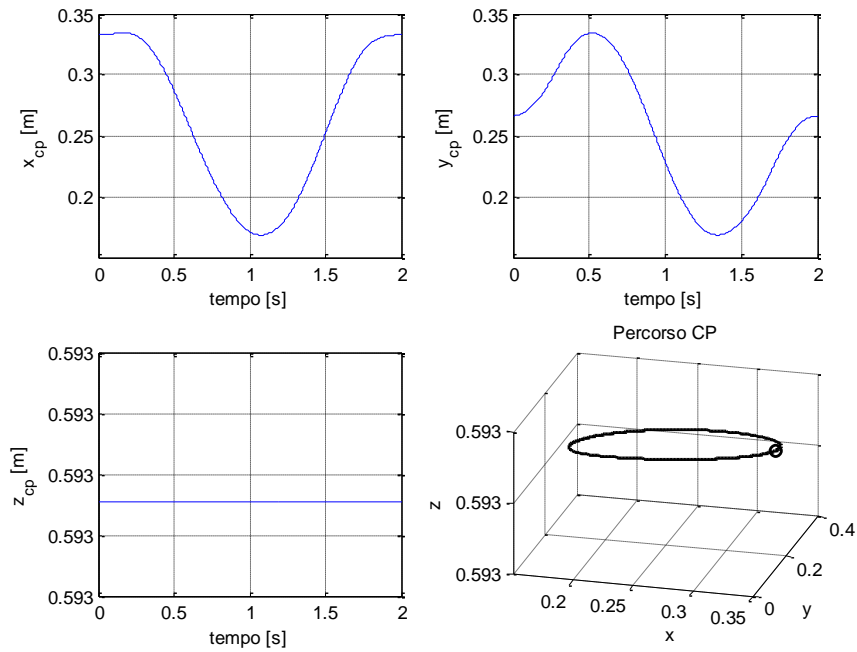


Figura 10: Traiettoria e percorso del centro polso CP

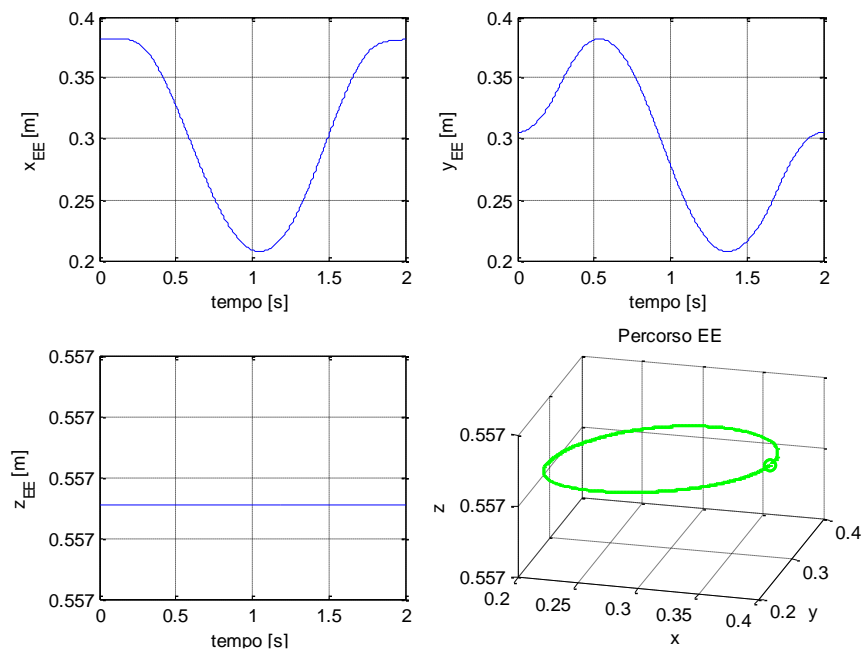


Figura 11: Traiettoria e percorso del centro polso EE

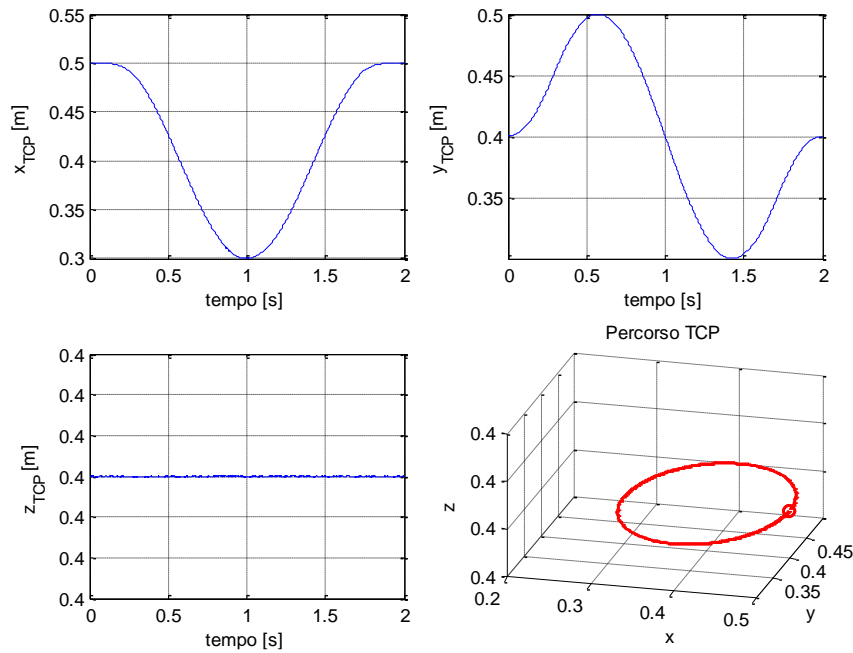


Figura 12: Traiettoria e percorso del tool centre point TCP

Il passo successivo è quello di calcolare il vettore generalizzato di velocità del *tool centre point*, il quale contiene l'informazione relativa alle velocità di traslazione e alle velocità angolari attorno agli assi. Pertanto, con l'utilizzo della matrice jacobiana (si veda paragrafo 9), si passa dallo spazio giunti a quello operativo:

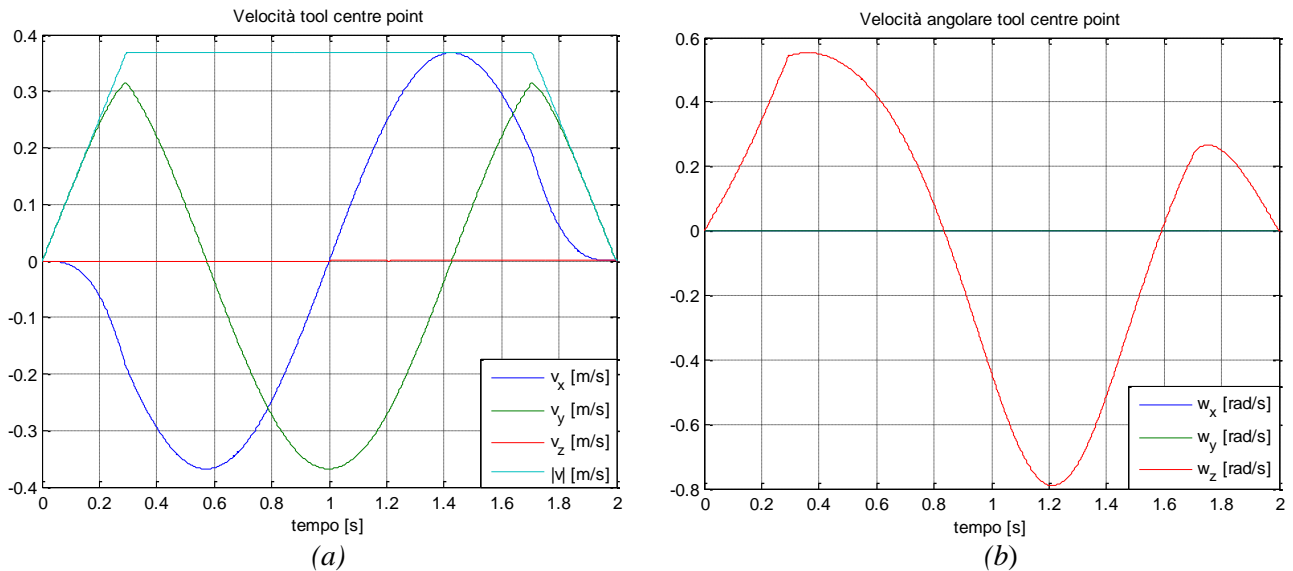


Figura 13: Velocità TCP lungo gli assi cartesiani (a), velocità angolari TCP (b)

Trattandosi di una traiettoria circolare su di un piano parallelo a  $\langle x_0 y_0 \rangle$ , la componente di velocità del TCP lungo l'asse  $z_{TPC}$  è nulla (figura 13.a); analogo discorso per le velocità angolari attorno agli assi  $x_{TPC}$  e  $y_{TPC}$  (figura 13.b).

Nel seguito è riportata una rappresentazione *wire-frame* del robot nelle configurazioni estreme e in due configurazioni intermedie.

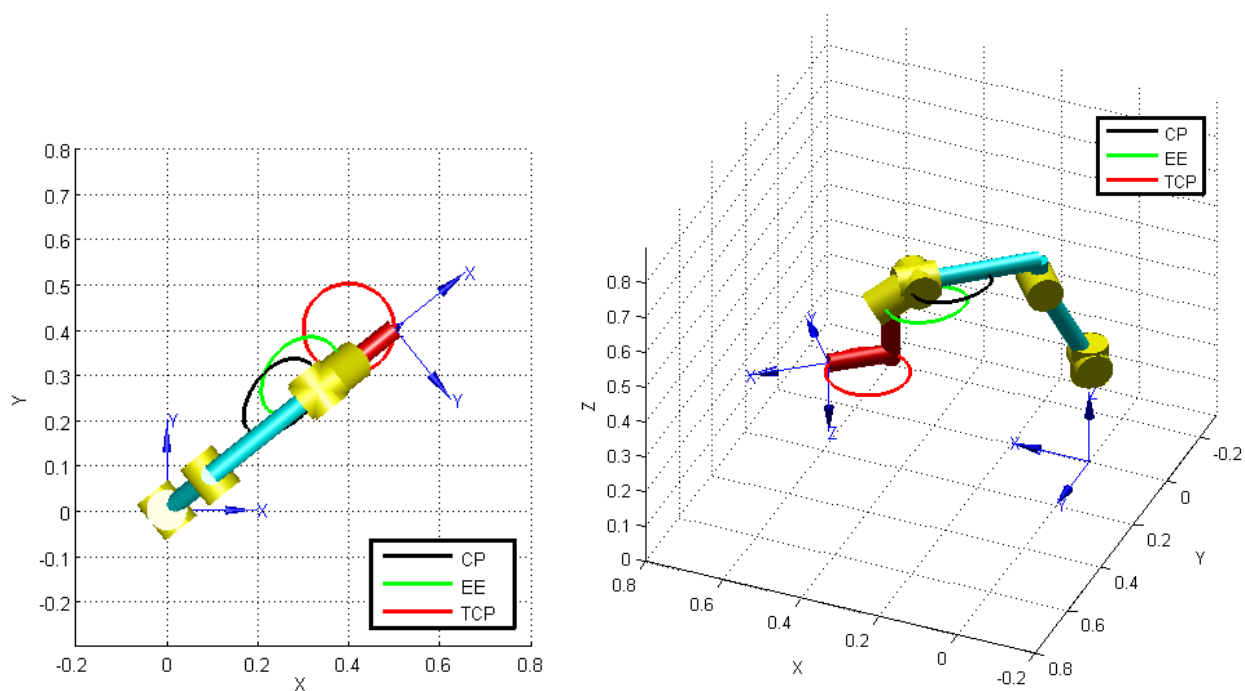


Figura 14: Rappresentazione wire-frame: configurazione iniziale

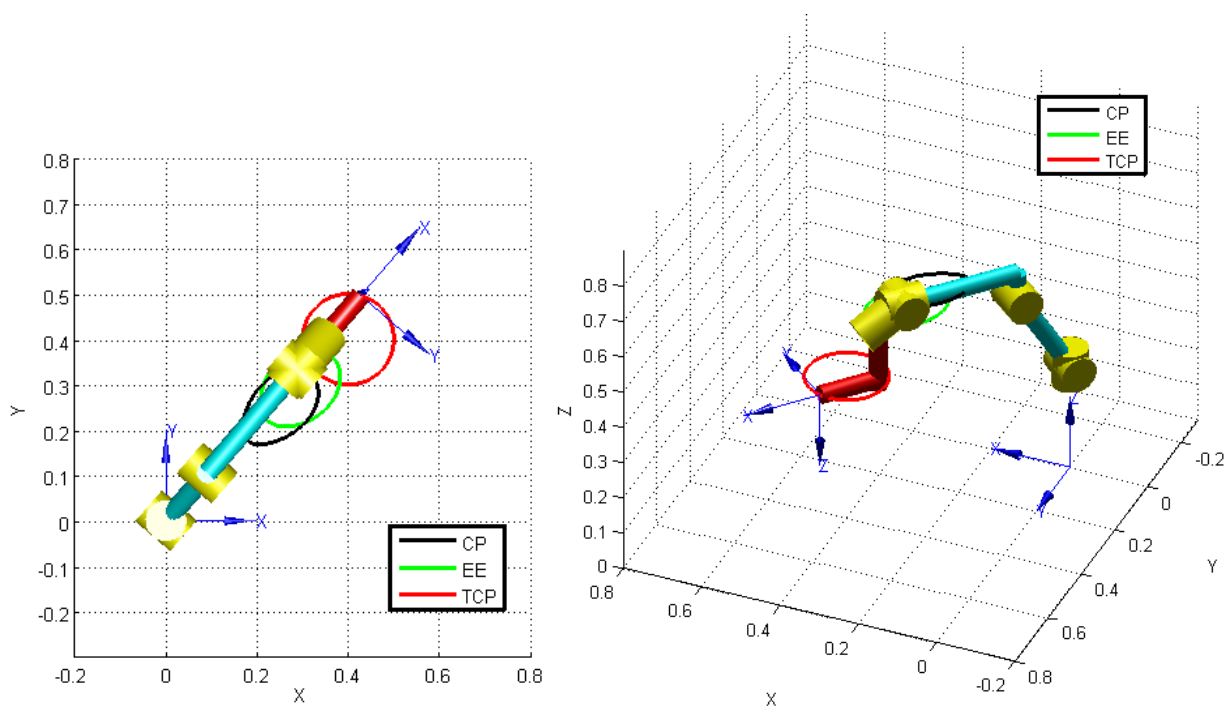


Figura 15: Rappresentazione wire-frame: configurazione  $1/4$  di traiettoria

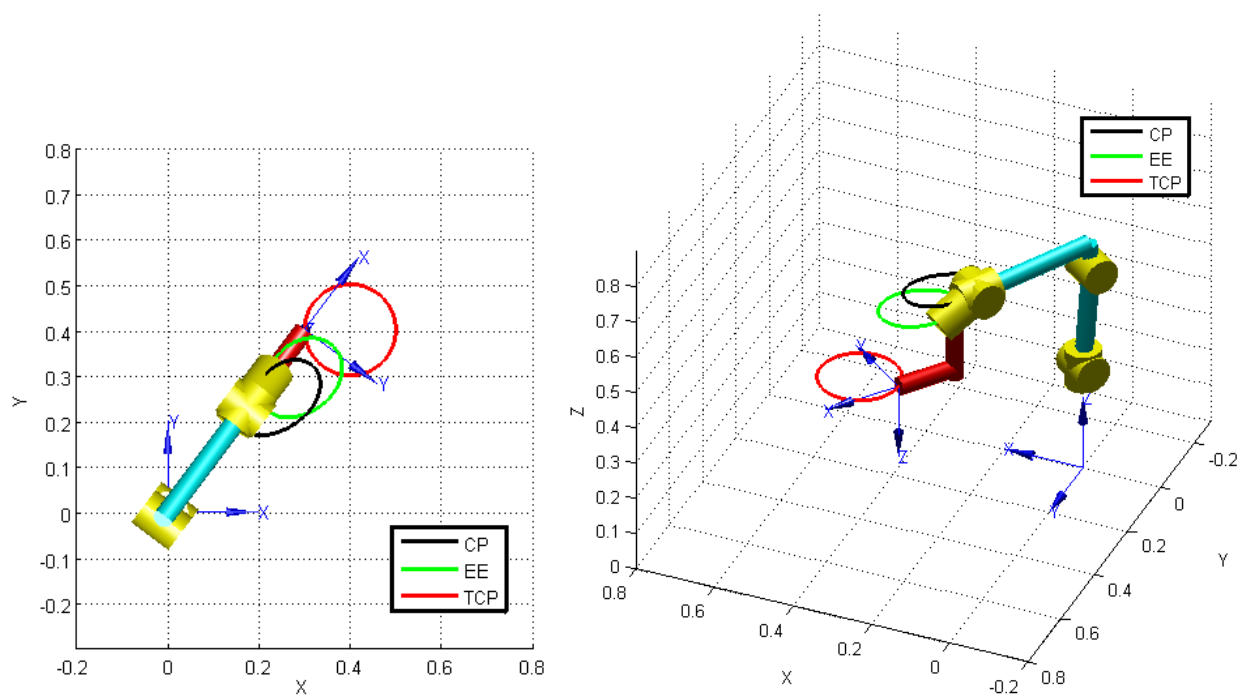


Figura 16: Rappresentazione wire-frame: configurazione  $\frac{1}{2}$  di traiettoria

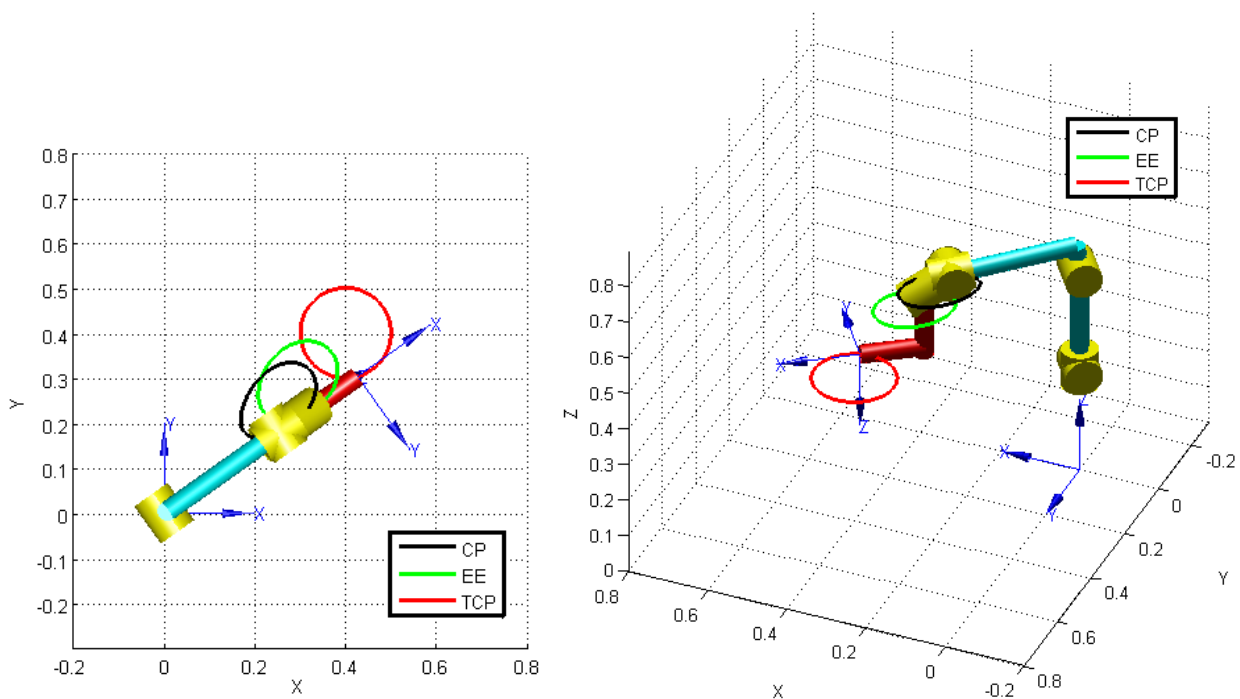


Figura 17: Rappresentazione wire-frame: configurazione  $\frac{3}{4}$  di traiettoria

#### 4. TENSORI DI INERZIA

Per poter affrontare un'analisi dinamica del robot occorre associare ad ogni link proprietà inerziali, quali:

- Massa;
- Posizione del baricentro rispetto al sistema di riferimento di D-H del corpo considerato;
- Tensore di inerzia baricentrico rispetto al sistema di riferimento di D-H del corpo considerato;

Inoltre, come si vedrà successivamente, si dovranno considerare anche le proprietà inerziali del *tool* e della base fissa: verranno dunque aggiunti all'oggetto robot (denominato nel listato "ABBirb120") creato con il comando "SerialLink", due link aggiuntivi rappresentativi di *tool* e base, creando così un oggetto (denominato "ABBirb120\_Dyn") avente 8 gradi di libertà complessivi.

Dai file CAD è possibile estrarre informazioni relative alle masse dei corpi, alle posizioni dei baricentri rispetto ai sistemi di riferimento CAD e ai tensori di inerzia centrali (ossia sia principali che baricentrici).

Per l'analisi successiva occorre convertire tali informazioni rispetto ai sistemi di riferimento di D-H, secondo le seguenti formulazioni:

$${}^{DHi}\hat{b}_i = {}^{DHi}\hat{A}_{CADi} * {}^{CADi}\hat{b}_i \quad 2)$$

Dove:

${}^{DHi}\hat{b}_i$  = vettore di posizione del baricentro del corpo i-esimo rispetto al sistema di riferimento DH i-esimo

${}^{DHi}\hat{A}_{CADi}$  = matrice omogenea (4x4) di trasformazione tra sistema di riferimento CAD i-esimo e sistema di riferimento DH i-esimo

${}^{CADi}\hat{b}_i$  = vettore di posizione del baricentro del corpo i-esimo rispetto al sistema di riferimento CAD i-esimo

$${}^{DHi}I_{Gi} = {}^{DHi}A_{CADi} * {}^{CADi}A_{CMi} * {}^{CMi}I_{Gi} * ({}^{DHi}A_{CADi} * {}^{CADi}A_{CMi})^T \quad 3)$$

Dove:

${}^{DHi}I_{Gi}$  = tensore di inerzia baricentrico del corpo i-esimo rispetto al sistema di riferimento DH i-esimo

${}^{DHi}A_{CADi}$  = minore (3x3) di orientazione tra sistema di riferimento CAD i-esimo e sistema di riferimento DH i-esimo

${}^{CADi}A_{CMi}$  = matrice (3x3) di orientazione tra sistema di riferimento centrale CM i-esimo (centrato nel baricentro e orientato secondo gli assi principali di inerzia) e sistema di riferimento CAD i-esimo

${}^{CMi}I_{Gi}$  = tensore di inerzia del corpi i-esimo centrale



- *Massa di ogni corpo [Kg]*

$$m_0 = 9.6212;$$

$$m_1 = 4.2544;$$

$$m_2 = 4.4219;$$

$$m_3 = 4.5834;$$

$$m_4 = 1.3419;$$

$$m_5 = 0.7582;$$

$$m_6 = 0.0189;$$

$$m_T = 0.4365;$$

Utilizzando le equazioni 2) e 3) sono stati ottenuti i seguenti risultati:

- *Posizione baricentri nei sistemi di riferimento locali (D-H) per ogni link [m]*

$$b_0 = [-0.0420400000000000 \quad 0.0000800000000000 \quad 0.0796400000000000]';$$

$$b_1 = [0.0001000000000000 \quad -0.0001200000000000 \quad -0.0515900000000000]';$$

$$b_2 = [0.1012400000000000 \quad 0.0007800000000000 \quad -0.0021200000000000]';$$

$$b_3 = [0.0579100000000000 \quad 0.0228100000000000 \quad 0.0010600000000000]';$$

$$b_4 = [0.0004100000000000 \quad -0.0001500000000000 \quad -0.0773000000000000]';$$

$$b_5 = [0.0000600000000000 \quad -0.0010900000000000 \quad 0.0000400000000000]';$$

$$b_6 = [0 \quad 0.0001700000000000 \quad -0.0070600000000000]';$$

$$b_T = [0.0113700000000000 \quad 0 \quad 0.0985700000000000]';$$

- *Tensori di inerzia baricentrici espressi nei sistemi di riferimento locali (D-H) per ogni link [kgm<sup>2</sup>]*

$$I_0 = \begin{bmatrix} 0.038278625395628 & 0 & -0.002034834160982 \\ 0 & 0.076053785670000 & 0 \\ -0.002034834160982 & 0 & 0.073123624354506 \end{bmatrix}$$

$I1 = 0.019722562158839 \quad 0.000017236341825 \quad 0.000031902553419$   
 $0.000017236341825 \quad 0.019981272543498 \quad -0.000027360582193$   
 $0.000031902553419 \quad -0.000027360582193 \quad 0.014498342423469$

$I2 = 0.029364036935908 \quad 0 \quad 0.000566619185909$   
 $0 \quad 0.068231040330000 \quad 0$   
 $0.000566619185909 \quad 0 \quad 0.047020282651678$

$I3 = 0.019770938976550 \quad -0.0022241668676450 \quad 0.000285039545019$   
 $-0.002224166867645 \quad 0.013011358441025 \quad 0.000125600803044$   
 $0.000285039545019 \quad 0.000125600803044 \quad 0.026022070753124$

$I4 = 0.005308952901592 \quad 0.000013609327313 \quad 0.000016496147110$   
 $0.000013609327313 \quad 0.004056184971450 \quad -0.000021621388326$   
 $0.000016496147110 \quad -0.000021621388326 \quad 0.002876737322661$

$I5 = 0.001131171140000 \quad 0 \quad 0$   
 $0 \quad 0.000561630640000 \quad 0$   
 $0 \quad 0 \quad 0.001238480600000$

$I6 = 1.0e-05 *$

$0.229267000000000 \quad 0 \quad 0$   
 $0 \quad 0.234318000000000 \quad 0$   
 $0 \quad 0 \quad 0.411712000000000$

Tali risultati sono stati in seguito associati ad ogni link del robot attraverso un metodo della funzione “Link”.

Come già accennato precedentemente, occorre creare due link aggiuntivi per la base e per il *tool*. E' bene sottolineare che a tali corpi saranno attribuiti gradi di libertà nulli, essendo la base solidale al pavimento e il *tool* all'*end effector* del robot.

```
% NUOVA COSTRUZIONE DEL ROBOT (AGGIUNTA LA BASE E IL TOOL)
%(AGGIUNGO LA BASE E IL TOOL PER POTERNE DEFINIRE LE PROPRIETA' INERZIALI)

alphaT=0; aT=0; dT=0; tetaT0=0; qlimT=[0 0]*pi/180;
alphaB=0; aB=0; dB=0; tetaB0=0; qlimB=[0 0]*pi/180;

%tool
T(1)=Link('alpha',alphaT,'a',aT,'d',dT,'offset',tetaT0,'modified','qlim',qlimT);
tool= SerialLink(T,'name','tool'); % to build up a SerialLink object

T(1).m = mT; % mass
T(1).r = bT; % position COG
T(1).I = dhIGT; % inertia tensor

%base
B(1)=Link('alpha',alphaB,'a',aB,'d',dB,'offset',tetaB0,'modified','qlim',qlimB);
base= SerialLink(B,'name','base'); % to build up a SerialLink object

B(1).m = m0; % mass
B(1).r = b0; % position COG
B(1).I = dhIG0; % inertia tensor

ABBBirb120_Dyn=SerialLink([base,ABBBirb120,tool],'name','ABBBirb120_D_y_n');
ABBBirb120_Dyn.tool=eeAtcp;
```

## 5. AZIONI MOTRICI

In tale sezione si vuole risolvere l'equilibrio dinamico del robot al fine di calcolare le azioni motrici richieste in ogni giunto per tutta la durata della traiettoria assegnata precedentemente.

Si suppone che, durante la traiettoria, agisca in direzione dell'asse dell'ugello una forza di reazione dovuta al flusso di fluido uscente dall'ugello di intensità costante pari a 30 N.

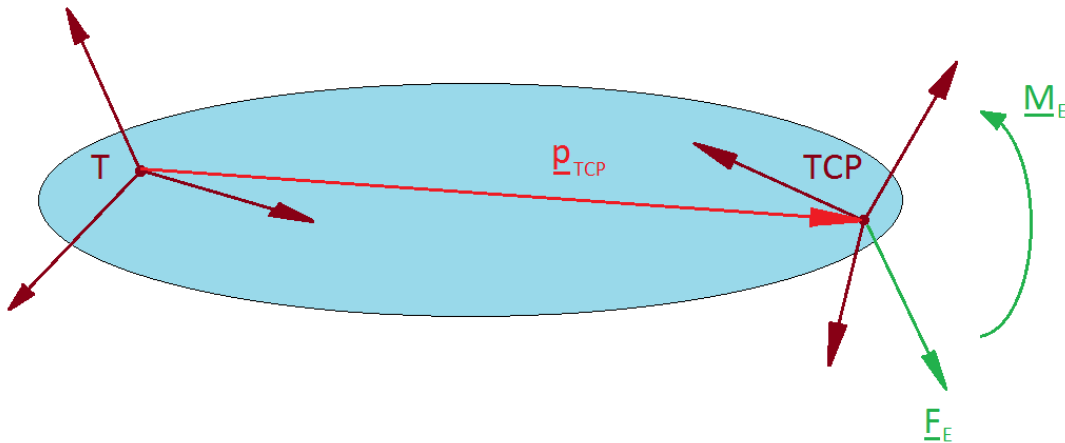
Pertanto, conoscendo il moto dei corpi del robot ("effetto") e volendo determinare le azioni sui corpi attraverso i giunti e con l'esterno ("causa"), si tratta di un problema di dinamica inversa.

Un metodo di risoluzione risiede nell'algoritmo **RNE** ("*Recursive Newton-Euler*"), implementato nella funzione "**rne\_mdh\_POLITO.m**":

```
[TAU,WBASE] = ABBirb120_Dyn.rne_mdh_POLITO([Q,QD,QDD],GRAV,FEXT')
```

Dove:

- TAU=azioni motrici richieste nei giunti per l'equilibrio dinamico
- WBASE=wrench esercitato sulla base per l'equilibrio dinamico
- Q=vettore valori gdl nei giunti (1x6 giunti)
- QD=vettore velocità gdl nei giunti (1x6 giunti)
- QDD=vettore accelerazione gdl nei giunti (1x6 giunti)
- GRAV=vettore accelerazione di gravità da assegnare alla base espresso nel sistema di riferimento di base per considerare i pesi propri dei link (GRAV=[0 0 9.81]'; %m/s2)
- FEXT=vettore di forze generalizzate (1x6) esercitate dal link N (ugello di verniciatura) sull'esterno, espresso nel sistema di riferimento secondo Denavit-Hartenberg del link stesso



$$\underline{F}_{-E} = \begin{Bmatrix} {}^{TCP}\underline{F}_E \\ {}^{TCP}\underline{M}_E \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ 30 \\ 0 \\ 0 \\ 0 \end{Bmatrix} \Rightarrow FEXT = \begin{Bmatrix} {}^T\underline{F}_{ET} \\ {}^T\underline{M}_{ET} \end{Bmatrix} = \begin{Bmatrix} {}^T A_{TCP} * {}^{TCP}\underline{F}_E \\ {}^T A_{TCP} * {}^{TCP}\underline{M}_E + {}^T \underline{p}_{TCP} \times {}^T A_{TCP} * {}^{TCP}\underline{F}_E \end{Bmatrix}$$

Come si può vedere dalle equazioni precedenti, il vettore di forze generalizzato riferito al *tool centre point* è stato ridotto al punto *T* coincidente con il punto *end effector*.

Un estratto del listato:

```
%% DINAMICA

q0vet=zeros(1001,1);
Q=deg2rad([q0vet,q1vet,q2vet,q3vet,q4vet,q5vet,q6vet,q0vet]);
QD=deg2rad([q0vet,q1pvet,q2pvet,q3pvet,q4pvet,q5pvet,q6pvet,q0vet]);
QDD=deg2rad([q0vet,q1ppvet,q2ppvet,q3ppvet,q4ppvet,q5ppvet,q6ppvet,q0vet]);

forza=30;
GRAV=[0;0;9.81];
tcpFe=[0,0,forza]';
tcpMe=[0,0,0]';

tFet=eeAtcp(1:3,1:3)*tcpFe;
tMet=eeAtcp(1:3,1:3)*tcpMe+cross(eeAtcp(1:3,4),eeAtcp(1:3,1:3)*tcpFe);

FEXT=[tFet; tMet];

for h=1:length(tempovet)

[TAU(h,:),WBASE(:,h)]=ABBirb120_Dyn.rne_mdh_POLITO([Q(h,:),QD(h,:),QDD(h,:)],...
GRAV,FEXT');

end
```

Nella figura 18, il risultato relativo alle azioni motrici nei giunti:

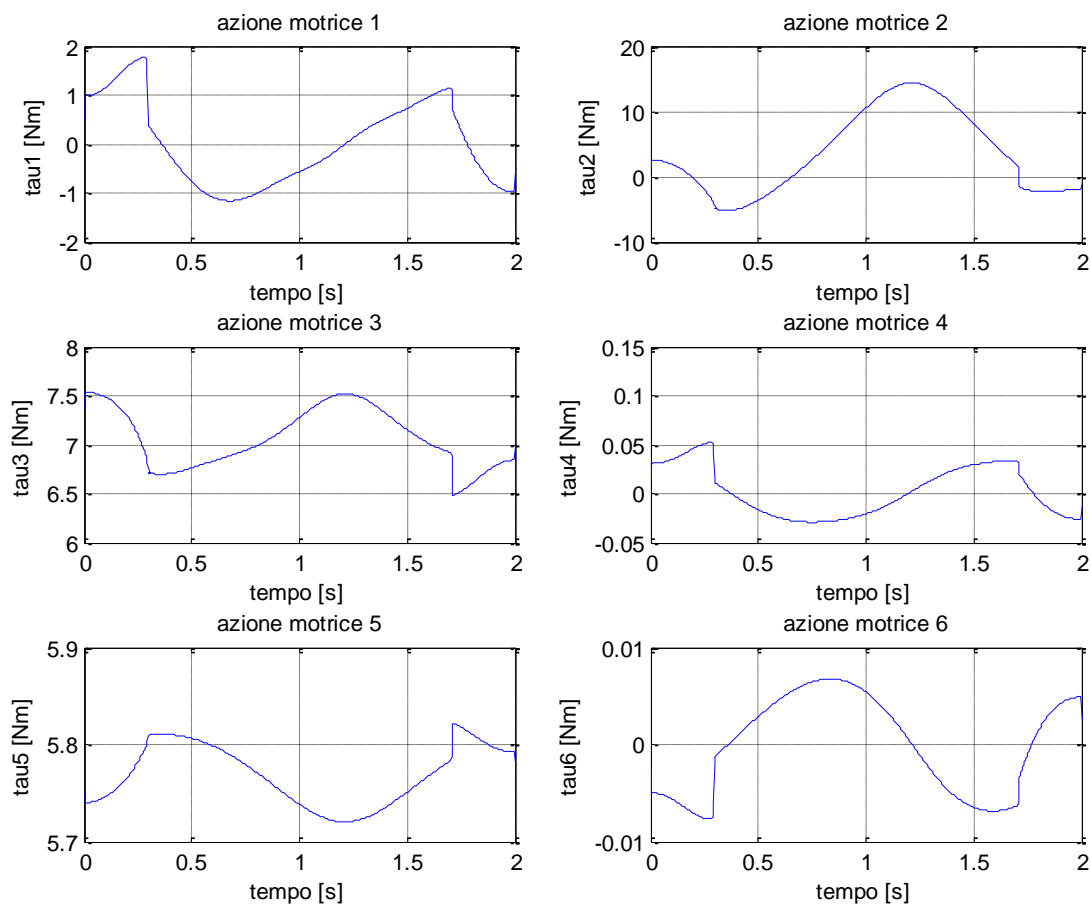
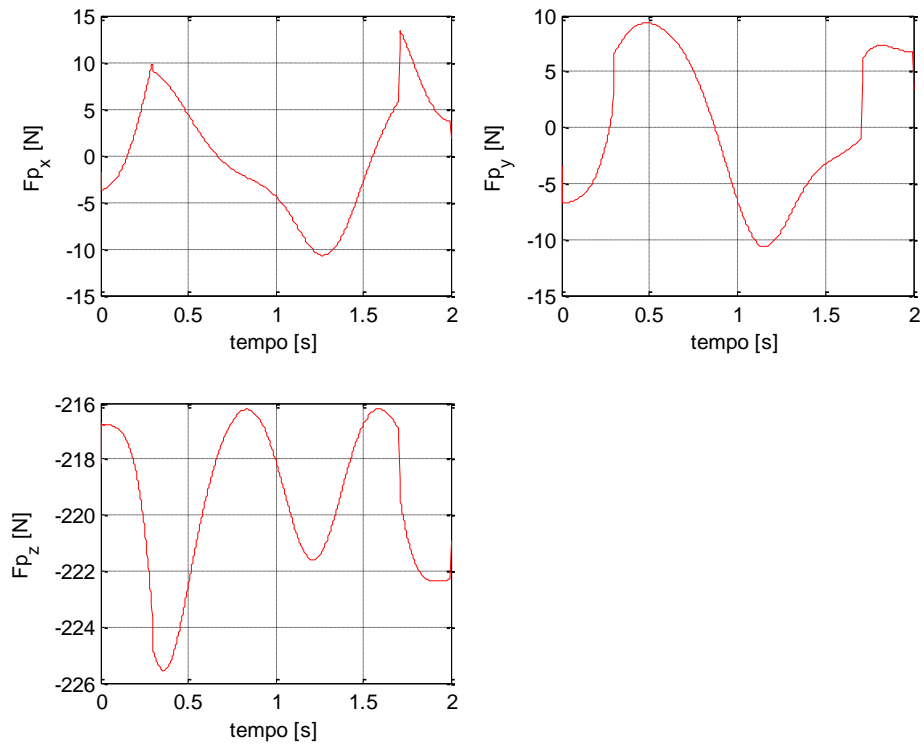
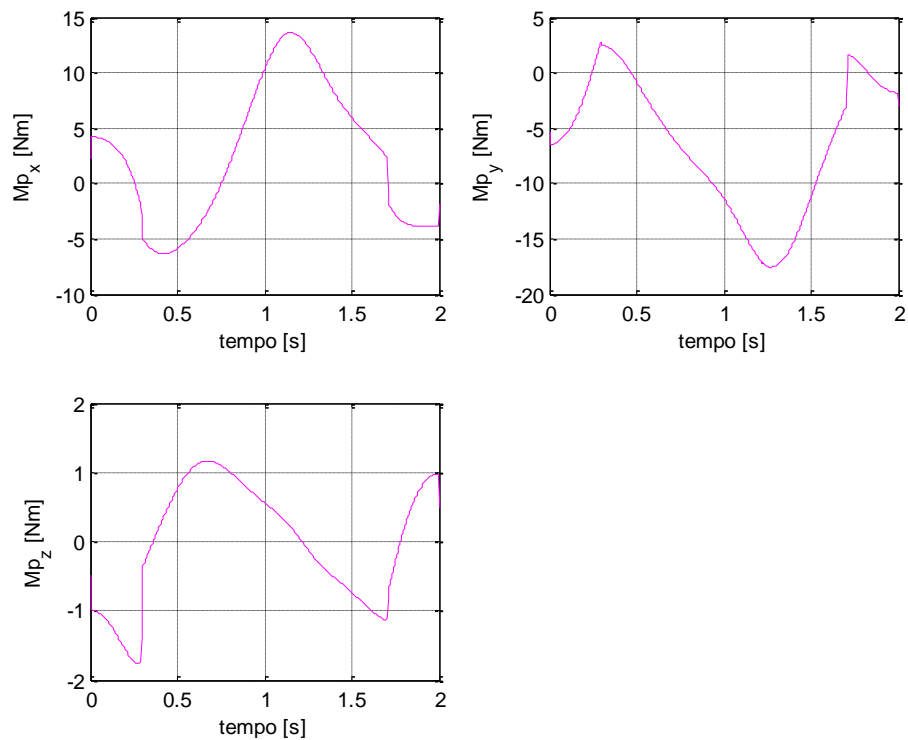


Figura 18: Azioni motrici nei giunti

La funzione utilizzata consente di ottenere anche il vettore generalizzato di forze scaricate a pavimento (“WBASE”). Gli andamenti nel tempo sono riportati nella figura 19.



a)



b)

Figura 19: Sollecitazioni trasmesse a pavimento ( a) Forze; b) Momenti)

## 6. SPAZIO DI LAVORO

Nell'analisi di un robot, un punto di fondamentale importanza risiede nello studio del suo spazio di lavoro. Avendo dei limiti definiti per i gradi di libertà nei giunti, sarà univocamente determinato il luogo geometrico dei punti raggiungibili dal robot. In tale sezione si fa riferimento al posizionamento del centro polso CP. Il catalogo propone la seguente mappatura:

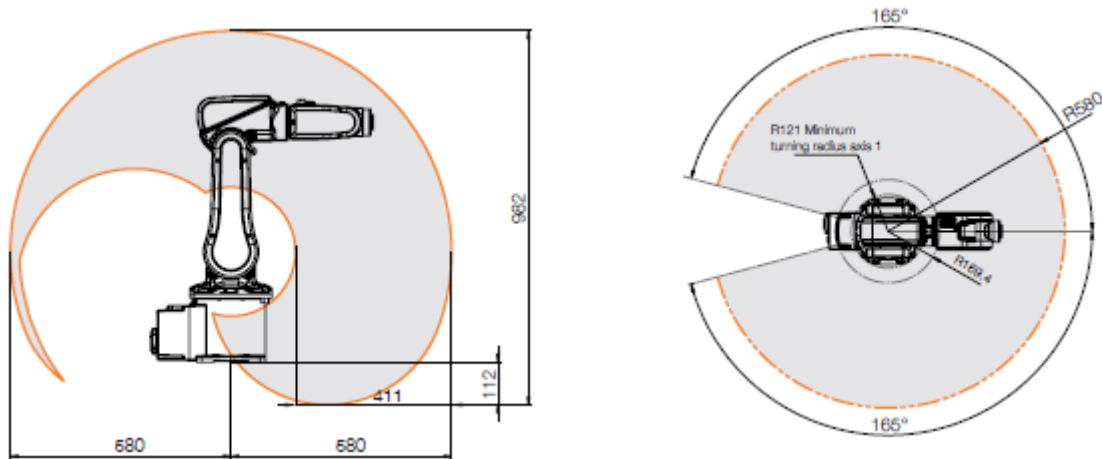


Figura 20: Mappatura spazio di lavoro da catalogo

Come si può notare dalla figura 20, ci sono configurazioni per cui il centro polso potrebbe posizionarsi in un punto inferiore al piano di appoggio del robot (possibile solo in caso di montaggio su piedistallo), o addirittura potrebbe compenetrare la base fissa; da qui la necessità di applicare opportune correzioni.

`%% MAPPATURA SPAZIO DI LAVORO`

```
q2vec=linspace(min(qlim2),max(qlim2),50);
q3vec=linspace(min(qlim3),max(qlim3),50);
```

```
i23=1;
for i2=1:length(q2vec)
    for i3=1:length(q3vec)
        q1=0;
        q2=q2vec(i2);
        q3=q3vec(i3);
        q4=0;
        q5=0;
        q6=0;
        zeroA4=ABBirb120.A([1 2 3 4],[q1 q2 q3 q4]); % Centro Polso
        zeroA6=ABBirb120.A([1 2 3 4 5 6],[q1 q2 q3 q4 q5 q6]); % End Effector
        C4=zeroA4(1:3,4);
        C6=zeroA6(1:3,4);
        wspace(i23,:)= [q1 q2 q3 q4 q5 q6 C4' C6' 0];
        i23=i23+1;
    end
end
```



```

% punti da escludere dallo spazio di lavoro potenziale del robot

for iCR=1:numrows(wspace)
    if wspace(iCR,9)<0 | wspace(iCR,9)<0.2 & abs(wspace(iCR,7))<0.1
        wspace(iCR,13)=1; % flag posto a uno per i valori critici
    end
end

wspace01=sortrows(wspace,13); % ordino le righe di wspace in modo che si abbiano
prima quelle con flag=0 e poi quelle con flag=1
row1=find(wspace01(:,13),1); % trovo la prima riga in cui si ha flag=1 e ne
salvo l'indice in row1
wspace0=wspace01(1:row1-1,:); % porzione del wspace contenente i punti
raggiungibili
wspace1=wspace01(row1:numrows(wspace01),:); % porzione del wspace contenente i
punti critici

```

Nella figura che segue, è rappresentato lo spazio di lavoro e sono evidenziate le criticità.

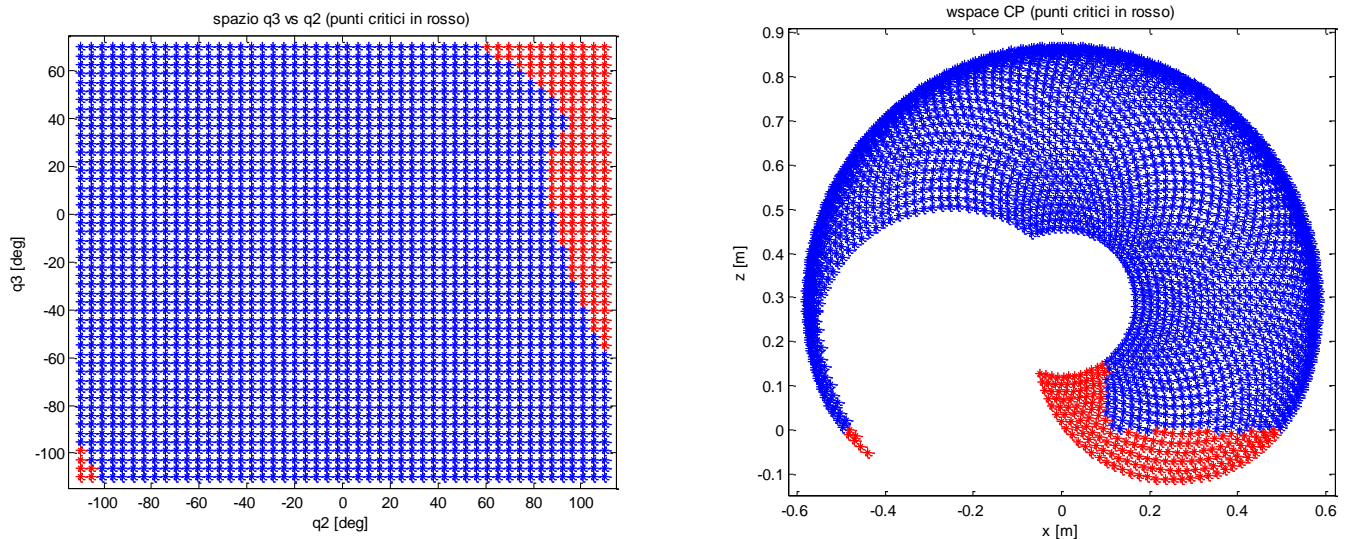


Figura 21: Mappatura spazio di lavoro con criticità

## 7. ALGORITMO DI CINEMATICA INVERSA

Nella cinematica diretta assegnato un vettore di gradi di libertà nei giunti è univocamente determinata la posa dell'*end effector*.

Il problema di cinematica inversa può essere risolto con metodi numerici oppure in forma chiusa.

Per la soluzione in forma chiusa (praticabile qualora si possa individuare un centro polso), ad una data posa dell'*end effector* corrispondono 8 possibili vettori di gradi di libertà nei giunti (escluse le configurazioni di singolarità).

Il punto centro polso permette di dividere il problema in due parti: cinematica inversa di braccio e cinematica inversa di polso (ciascuna a tre gradi di libertà).

Calcolo posizione centro polso CP (coincidente con  $O_4$ ) w.r.t. S.R.0, nota  ${}^0\hat{A}_6$

$${}^0\underline{p}_4 = {}^0\underline{p}_6 - d_6 {}^0\underline{k}_6 = [x_4 \quad y_4 \quad z_4]^t$$

Calcolo posizione centro polso CP w.r.t. S.R.1

$$1) \quad {}^1\underline{\hat{p}}_4 = {}^1\hat{A}_2 {}^2\hat{A}_3 {}^3\underline{\hat{p}}_4 = \begin{bmatrix} s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ c_2 & -s_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_3 & -s_3 & 0 & a_2 \\ s_3 & c_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_3 \\ d_4 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} s_{23}a_3 + c_{23}d_4 + s_2a_2 \\ 0 \\ c_{23}a_3 - s_{23}d_4 + c_2a_2 \\ 1 \end{bmatrix}$$

$$2) \quad {}^1\underline{\hat{p}}_4 = {}^0\hat{A}_1^{-1} {}^0\underline{\hat{p}}_4 = \begin{bmatrix} c_1 & s_1 & 0 & 0 \\ -s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & -d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_4 \\ y_4 \\ z_4 \\ 1 \end{bmatrix} = \begin{bmatrix} c_1x_4 + s_1y_4 \\ -s_1x_4 + c_1y_4 \\ z_4 - d_1 \\ 1 \end{bmatrix}$$

Risoluzione cinematica inversa braccio

$$\begin{cases} c_1x_4 + s_1y_4 = s_{23}a_3 + c_{23}d_4 + s_2a_2 & (1) \\ -s_1x_4 + c_1y_4 = 0 & (2) \\ z_4 - d_1 = c_{23}a_3 - s_{23}d_4 + c_2a_2 & (3) \end{cases}$$

da equazione (2):

$$\begin{cases} l = \sqrt{x_4^2 + y_4^2} \\ \gamma = \text{atan2}(y_4, x_4) \end{cases} \quad \begin{cases} x_4 = lc_\gamma \\ y_4 = ls_\gamma \end{cases}$$

$$(-s_1c_\gamma + c_1s_\gamma)l = 0$$

$$\sin(\gamma - q_1)l = 0$$

$$\gamma - q_1 = \text{atan2}(0, \pm 1) \quad \text{per } l \neq 0$$

$$q_1 = \text{atan2}(y_4, x_4) - \text{atan2}(0, \pm 1)$$

Sommando il quadrato delle tre equazioni:

$$\begin{aligned} x_4^2 + y_4^2 + z_4^2 + d_1^2 - 2d_1z_4 &= a_3^2 + d_4^2 + a_2^2 + 2a_3a_2c_3 - 2d_4a_2s_3 \\ a_3c_3 - d_4s_3 &= \frac{x_4^2 + y_4^2 + z_4^2 + d_1^2 - 2d_1z_4 - a_3^2 - d_4^2 - a_2^2}{2a_2} = B \end{aligned}$$

$$\begin{cases} e = \sqrt{a_3^2 + d_4^2} \\ \xi = \text{atan2}(a_3, d_4) \end{cases} \quad \begin{cases} a_3 = e s_\xi \\ d_4 = e c_\xi \end{cases}$$

$$(s_\xi c_3 - c_\xi s_3) e = B$$

$$\begin{cases} \sin(\xi - q_3) = \frac{B}{e} \\ \cos(\xi - q_3) = \pm \sqrt{1 - \left(\frac{B}{e}\right)^2} \end{cases} \quad \text{per } e \neq 0$$

$$\xi - q_3 = \text{atan2}\left(B, \pm \sqrt{e^2 - B^2}\right)$$

$$q_3 = \text{atan2}(a_3, d_4) - \text{atan2}\left(B, \pm \sqrt{a_3^2 + d_4^2 - B^2}\right)$$

Combinando le equazioni (1) e (2)

$$\begin{cases} c_1x_4 + s_1y_4 = s_2[c_3a_3 - s_3d_4 + a_2] + c_2[s_3a_3 + c_3d_4] \\ z_4 - d_1 = -s_2[s_3a_3 + c_3d_4] + c_2[c_3a_3 - s_3d_4 + a_2] \end{cases}$$

$$\begin{cases} s_2 = \frac{(c_1x_4 + s_1y_4)(c_3a_3 - s_3d_4 + a_2) - (z_4 - d_1)(s_3a_3 + c_3d_4)}{(c_3a_3 - s_3d_4 + a_2)^2 + (s_3a_3 + c_3d_4)^2} \\ c_2 = \frac{(c_1x_4 + s_1y_4)(s_3a_3 + c_3d_4) + (z_4 - d_1)(c_3a_3 - s_3d_4 + a_2)}{(c_3a_3 - s_3d_4 + a_2)^2 + (s_3a_3 + c_3d_4)^2} \end{cases}$$

$$q_2 = \text{atan2}(s_2, c_2)$$

4 soluzioni della cinematica inversa braccio (escluse configurazioni di singolarità):

1.  $\{q_1^I \ q_2^I \ q_3^I\}$
2.  $\{q_1^I \ q_2^{II} \ q_3^{II}\}$
3.  $\{q_1^{II} \ q_2^{III} \ q_3^I\}$
4.  $\{q_1^{II} \ q_2^{IV} \ q_3^{II}\}$

Risoluzione cinematica inversa polso

Matrici note:

$${}^0\hat{A}_3 = {}^0\hat{A}_1 {}^1\hat{A}_2 {}^2\hat{A}_3$$

$${}^3\hat{A}_6 = {}^0\hat{A}_3^{-1} {}^0\hat{A}_6$$

Espressione letterale di  ${}^3\hat{A}_6$ :

$${}^3\hat{A}_6 = {}^3\hat{A}_4 {}^4\hat{A}_5 {}^5\hat{A}_6 = \begin{bmatrix} c_4c_5c_6 - s_4s_6 & -c_4c_5s_6 - s_4c_6 & -c_4s_5 & -c_5s_5d_6 + a_3 \\ s_5c_6 & -s_5s_6 & c_5 & c_5d_6 + d_4 \\ -s_4c_5c_6 - c_4s_6 & s_4c_5s_6 - c_4c_6 & s_4s_5 & s_4s_5d_6 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

indicando  ${}^3A_6 = [A_{ij}]$  i gdl del polso sono:

$$q_5 = \pm \cos^{-1}(A_{23})$$

$$q_4 = \text{atan2}(A_{33}/s_5, -A_{13}/s_5)$$

per  $s_5 \neq 0^\circ; 180^\circ$

$$q_6 = \text{atan2}(-A_{22}/s_5, A_{21}/s_5)$$

$$A_{23} = +1 \Rightarrow q_5 = 0^\circ \Rightarrow q_4 + q_6 = \text{atan2}(-A_{12}, A_{11})$$

$$A_{23} = -1 \Rightarrow q_5 = 180^\circ \Rightarrow q_4 - q_6 = \text{atan2}(-A_{12}, -A_{11}) \quad \text{per } s_5 = 0^\circ$$

2 soluzioni della cinematica inversa polso (escluse configurazioni di singolarità):

$$1. \{q_4^I \ q_5^I \ q_6^I\}$$

$$2. \{q_4^{II} \ q_5^{II} \ q_6^{II}\}$$

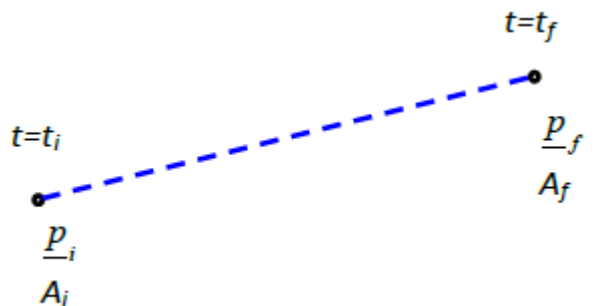
Questo procedimento è stato sviluppato nel listato “cin\_inversa.m”, riportato nell'allegato 4.

## 8. PIANIFICAZIONE TRAIETTORIA RETTILINEA TOOL CENTRE POINT TCP

In questa sezione si vuole pianificare una traiettoria di posizione dell'ugello di verniciatura con percorso rettilineo.

L'orientazione si vuole mantenere costante durante il movimento (non si effettua pianificazione della traiettoria di orientazione), quindi il risultato è una traslazione del *tool*.

La traiettoria di posizione è stata definita con logica di tipo “*point to point*”, si necessita quindi di una posa iniziale e finale; pertanto le condizioni agli estremi della traiettoria sono:

$t=t_i$ 	<div style="display: flex; flex-direction: column; align-items: center;"> <div>posizione</div> <div>orientazione</div> </div>	$\underline{p}(t_i) = \underline{p}_i; \quad \underline{p}(t_f) = \underline{p}_f$ $A(t_i) = A_i; \quad A(t_f) = A_f$
--	---	---

Si procede con la pianificazione della traiettoria di posizione:

percorso lineare:  $\underline{p}(s) = \underline{p}_i + s(\underline{p}_f - \underline{p}_i)$

traiettoria:  $\underline{p}(t) = \underline{p}_i + s(t)(\underline{p}_f - \underline{p}_i)$

Dove il parametro di curva  $s$  è una variabile scalare che permette di definire la percentuale di completamento del percorso (il parametro varia fra 0 e 1), scegliendo quindi una legge di variazione temporale per  $s$  si passa dal percorso alla traiettoria. In particolare si è adottata una legge

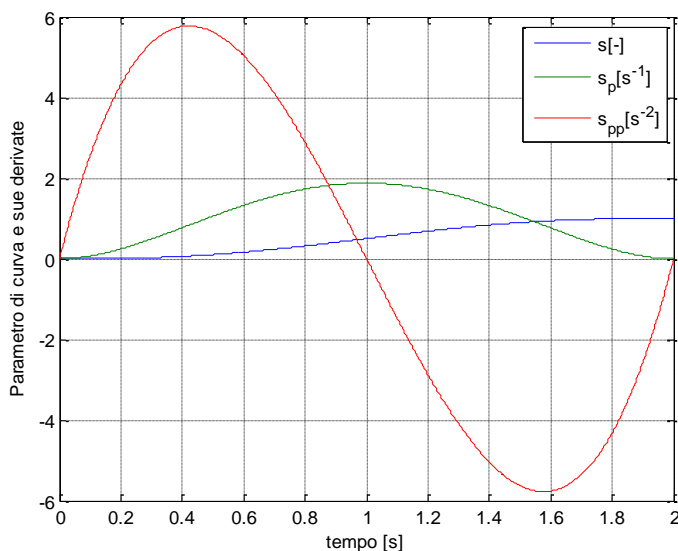


Figura 22: Legge polinomiale di quinto grado

polinomiale di quinto grado. Questa consente di imporre 6 condizioni (tante quanti sono i coefficienti del polinomio), numero minimo necessario per avere continuità anche sull'accelerazione.

Nella figura 22 si riportano gli andamenti temporali del parametro  $s$  e delle sue derivate (ottenuti mediante la *function* “*forma\_polinomiale345.m*” presente nell'allegato 5); inoltre è possibile notare che la traiettoria è pianificata su un intervallo di 2 secondi.

La posa iniziale dell'ugello di verniciatura è quella che si ottiene in configurazione di zero assi (per evitare di partire da una singolarità di polso, questa posa è stata ottenuta con una configurazione equivalente). Si è quindi stabilita la posa finale imponendo i seguenti spostamenti lungo gli assi del sistema di riferimento zero (fisso):

$$\Delta x = -0.08 \text{ m}; \Delta y = 0.20 \text{ m}; \Delta z = 0.03 \text{ m}.$$

Si riporta un estratto del listato (allegato 6):

```
%% SPECIFICHE TRAIETTORIA RETTILINEA

deltax=-0.08;
deltay=0.20;
deltaz=0.03;
spost_compl=sqrt(deltax^2+deltay^2+deltaz^2);
teta=asin(deltaz/spost_compl);
gamma=asin(deltay/(spost_compl*cos(teta)));
[s,s_p,s_pp,tempo]=forma_polinomiale345;

%% POSA TOOL PER OGNI ISTANTE DI TEMPO

zeroAtcp_tempo(:, :, 1)=zeroAtcp;
INVE(:, :, 1)=inv(eeAtcp);
zeroA6(:, :, 1)=zeroAtcp_tempo(:, :, 1)*INVE(:, :, 1);

for i=1:(length(tempo)-1)
    zeroAtcp_tempo(:, :, i+1)=zeroAtcp;
    zeroAtcp_tempo(:, 4, i+1)=zeroAtcp(:, 4)+s(i)*[deltax deltay deltaz 0]';
    INVE(:, :, i+1)=inv(eeAtcp);
    zeroA6(:, :, i+1)=zeroAtcp_tempo(:, :, i+1)*INVE(:, :, i+1);
end
```

Ottenute le 1001 pose in cui sono stati discretizzati i 2 secondi di traiettoria, è stato utilizzato un algoritmo di cinematica inversa per ottenere i vettori di gradi di libertà che potessero originarle. Per ognuno dei 1001 istanti si sono ottenute 8 possibili vettori di gradi di libertà (essendo che la cinematica inversa prevede 8 soluzioni). Tra un istante temporale e il successivo si è scelto di volta in volta il vettore di gradi di libertà che comportasse i minori movimenti nei giunti del robot.

```
%% CALCOLO DELLE 8 COMBINAZIONI PER PER OGNI ISTANTE DI TEMPO (2s)

for i=1:length(tempo)
    sol6gdl(:, :, i)=cin_inversa(zeroA6(:, :, i));
end

%% CERNITA DEI VETTORI Q
RIGAESATTA=2; % all'istante iniziale si sceglie manualmente una soluzione che
%eviti la singolarità di polso
GRADI_LIBERTA(1, :)=sol6gdl(RIGAESATTA, :, 1);
for i=1:(length(tempo)-1)
    for j=1:8
        vettore_norme(j)=norm(sol6gdl(j, :, i+1)-sol6gdl(RIGAESATTA, :, i));
    end
    [val index]=min(vettore_norme);
    RIGAESATTA=index;
    GRADI_LIBERTA(i+1, :)=sol6gdl(RIGAESATTA, :, i+1);
end
```

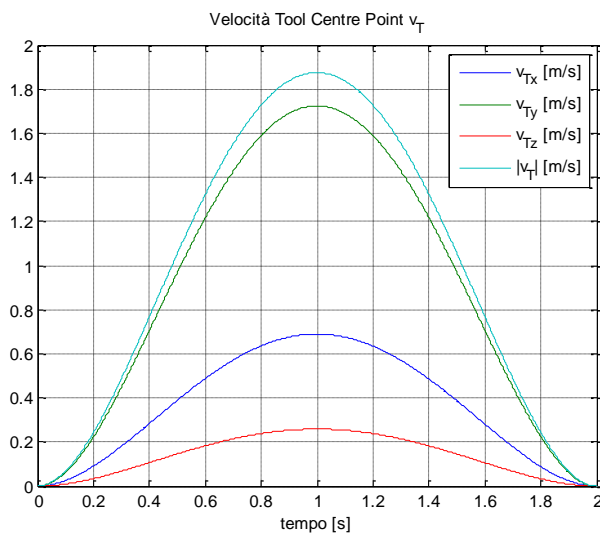
```
%% VETTORI GRADI DI LIBERTÀ DEI GIUNTI NEL TEMPO
```

```
q1=GRADI_LIBERTA(:,1);
q2=GRADI_LIBERTA(:,2);
q3=GRADI_LIBERTA(:,3);
q4=GRADI_LIBERTA(:,4);
q5=GRADI_LIBERTA(:,5);
q6=GRADI_LIBERTA(:,6);
```

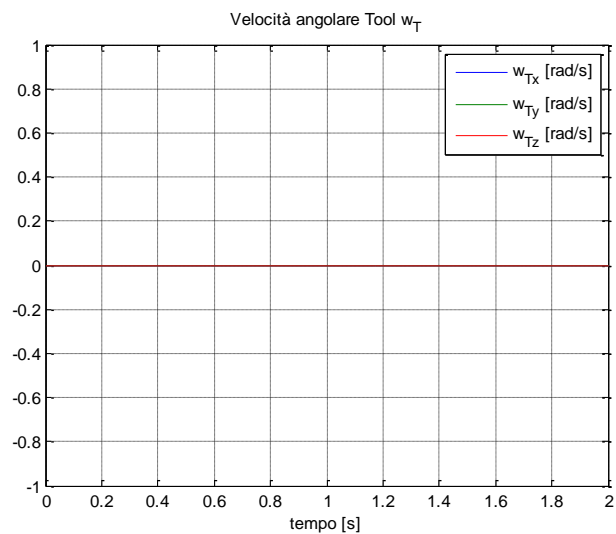
```
%% CALCOLO DELLE JACOBIANA E DELLE VELOCITÀ QP
```

```
for i=1:length(tempo)
    J(:, :, i)=ABBirb120.jacob0([q1(i) q2(i) q3(i) q4(i) q5(i) q6(i)]);
    V(:, i)=[s_p(i)*cos(teta)*cos(gamma); s_p(i)*cos(teta)*sin(gamma);
    s_p(i)*sin(teta); 0; 0; 0];
    qp(:, i)=inv(J(:, :, i))*V(:, i);
end
```

Scomponendo la derivata prima del parametro di curva nelle direzioni cartesiane, è possibile ottenere le velocità di traslazione del *tool centre point*; le velocità angolari invece sono nulle, essendo costante l'orientazione.



a)



b)

Figura 23: Velocità di traslazione TCP (a), velocità angolari TCP (b)

Nelle figure 24, 25 e 26 è riportata la traiettoria e il percorso del centro polso, dell'*end effector* e del *tool centre point*.

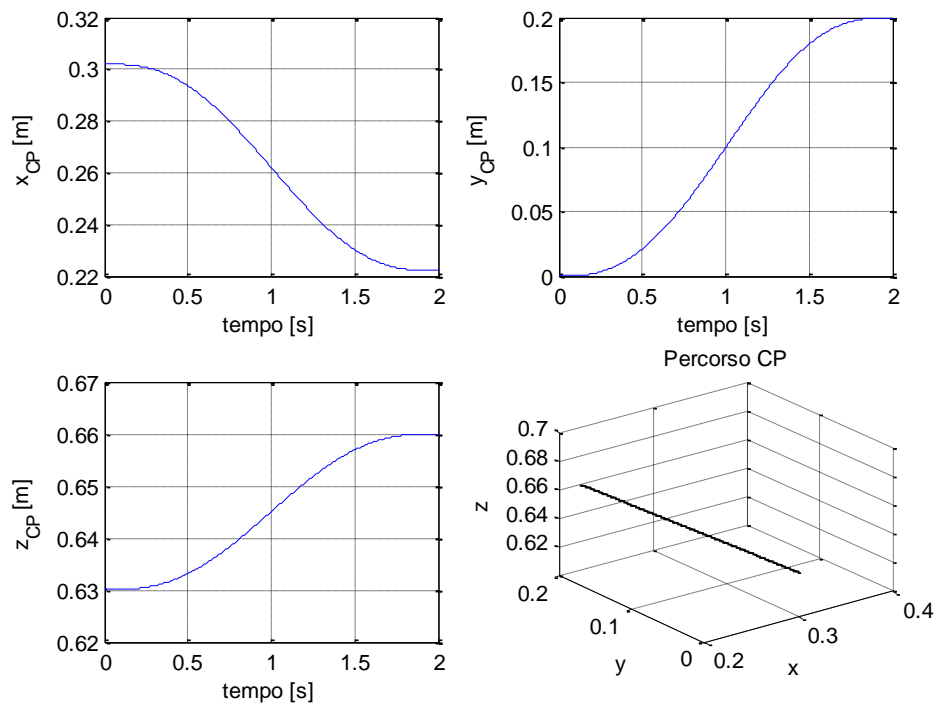


Figura 24: Traiettoria e percorso del centro polso CP

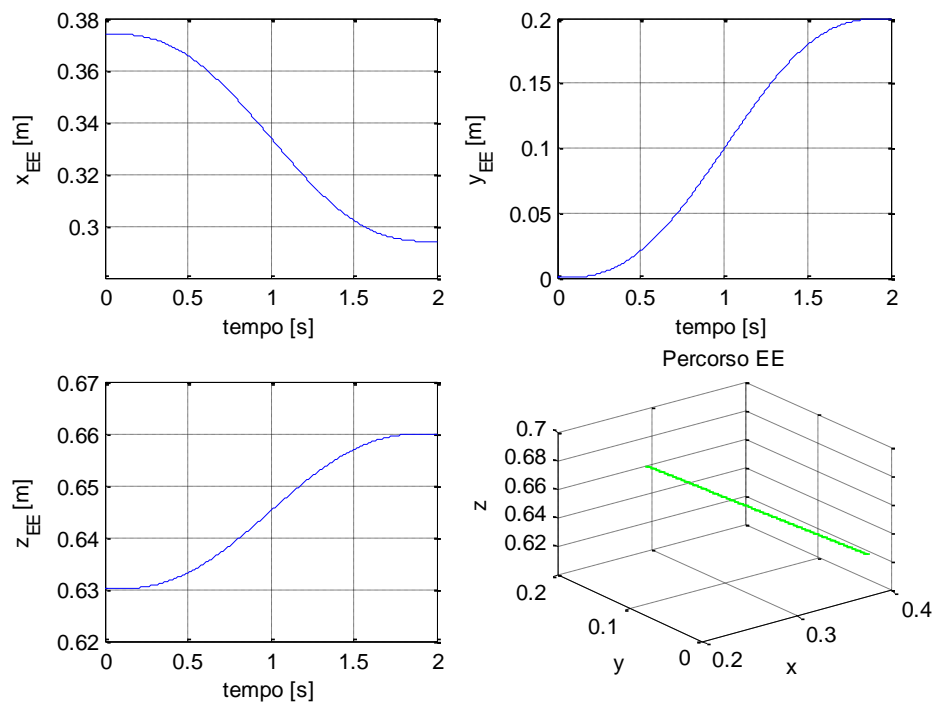


Figura 25: Traiettoria e percorso dell'*end effector* EE



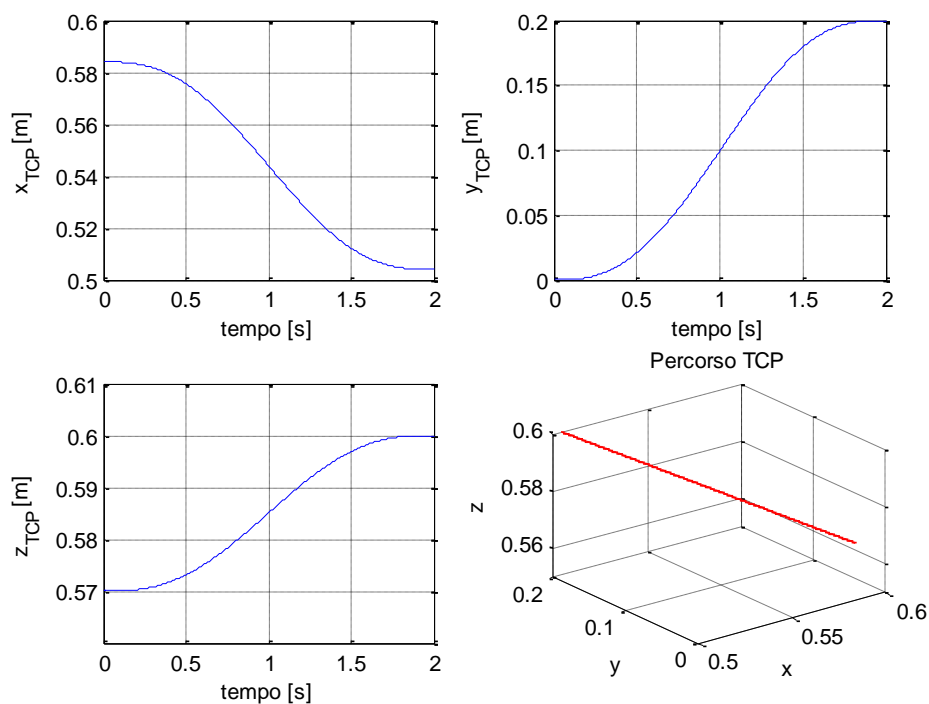


Figura 26: Traiettoria e percorso del tool centre point TCP

Nel seguito è riportata una rappresentazione *wire-frame* del robot nelle configurazioni estreme e in una configurazione intermedia.

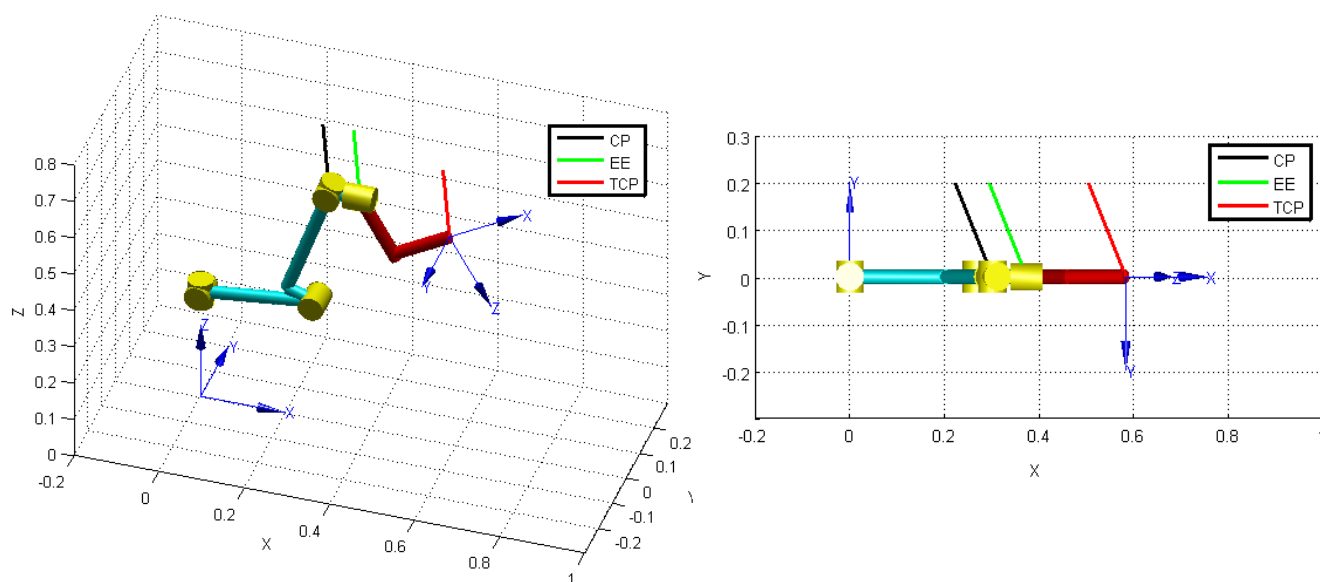
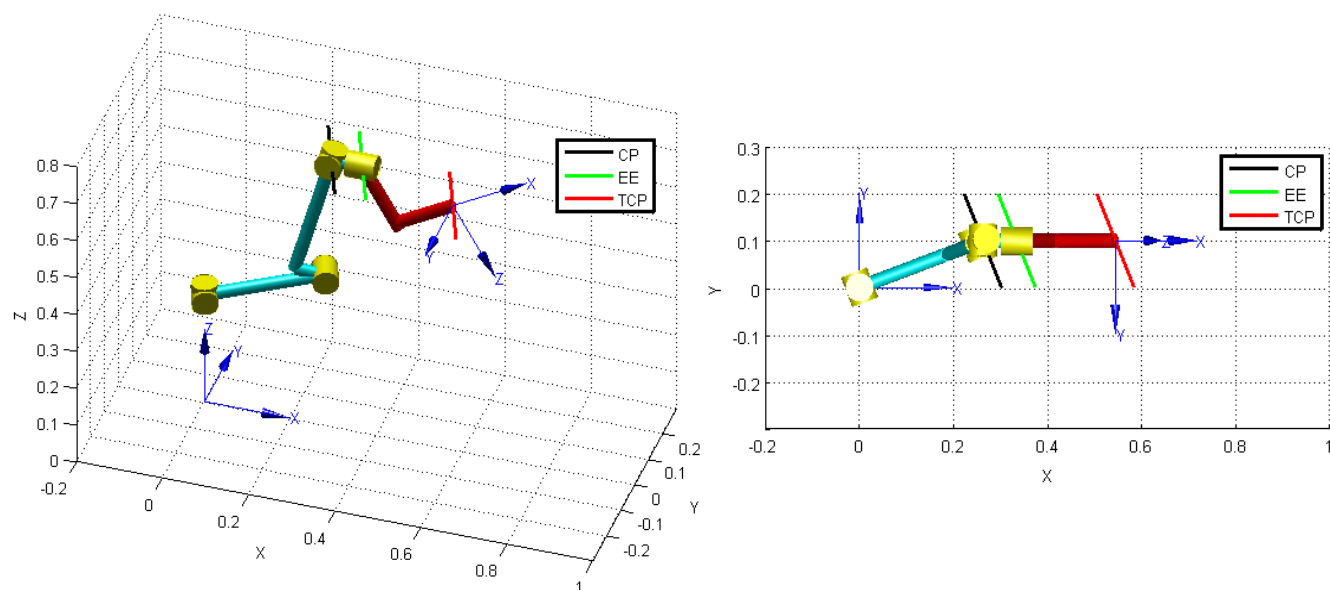
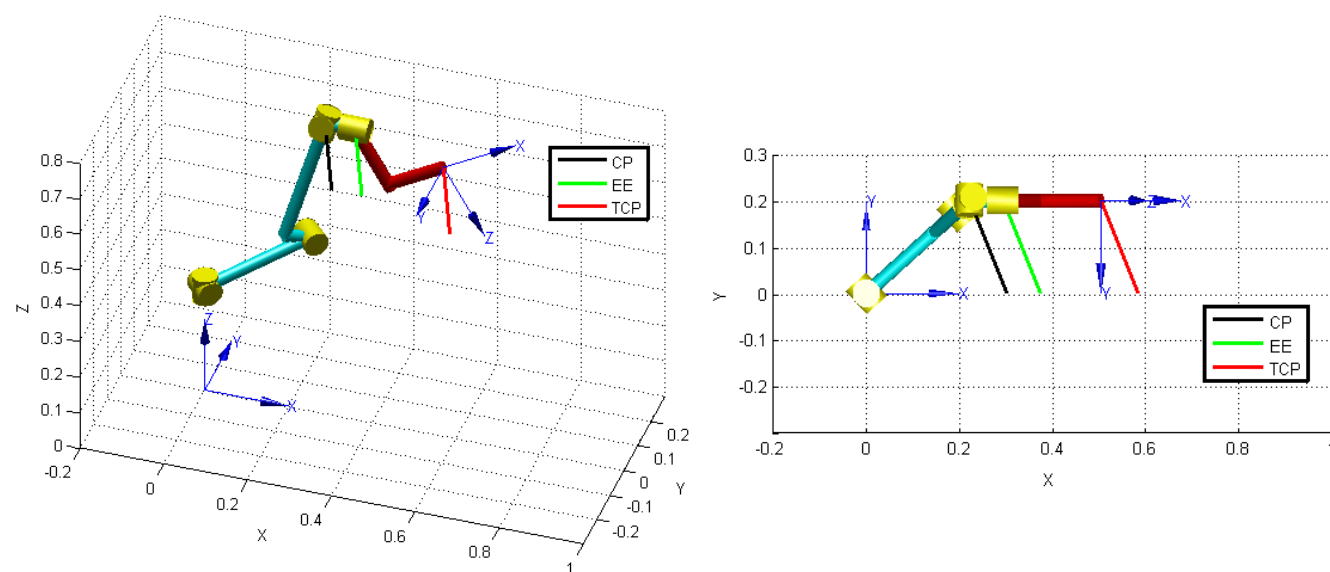


Figura 27: Configurazione iniziale

*Figura 28: Configurazione intermedia**Figura 29: Configurazione finale*

## 9. ANALISI CONFIGURAZIONI DI SINGOLARITA'

La matrice jacobiana stabilisce il legame cinematico tra le velocità nello spazio giunti e quelle nello spazio operativo.

In cinematica diretta, noto il vettore delle velocità dei gradi di libertà nei giunti, la velocità generalizzata del link n vale:

$$\underline{V}_n = J \underline{\dot{q}} \quad \left\{ \begin{array}{c} \underline{v}_n \\ \underline{\omega}_n \end{array} \right\} = [J_1 \mid J_2 \mid \dots \mid J_i \mid \dots \mid J_n] \left\{ \begin{array}{c} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \dot{q}_i \\ \vdots \\ \dot{q}_n \end{array} \right\}$$

Dove:

$$J_i = \begin{bmatrix} \delta_i \underline{k}_i + (1 - \delta_i) \underline{k}_i \times (\underline{p}_n - \underline{p}_i) \\ (1 - \delta_i) \underline{k}_i \end{bmatrix} \quad \begin{matrix} 3 \times 1 \\ 3 \times 1 \end{matrix}$$

Nella matrice jacobiana la j-esima riga (con j che varia da 1 a 6) rappresenta il contributo che ognuno dei gradi di libertà ha nel produrre la j-esima componente del vettore generalizzato di velocità.

Le colonne invece costituiscono i pesi che il grado di libertà i-esimo (con i che varia da 1 a n) ha nel generare ciascuna delle componenti del vettore generalizzato di velocità.

Inoltre, la matrice è funzione esclusivamente della configurazione di un dato robot.

A seconda del numero di gradi di libertà, il robot può essere definito “*sotto attuato*” o “*sovra attuato*”.

Nel primo caso il robot ha un numero di gradi di libertà inferiore al numero di componenti del vettore generalizzato di velocità (sei); la matrice jacobiana presenterà alcune righe linearmente dipendenti, pertanto, alcune componenti del vettore di velocità nello spazio operativo saranno dipendenti da altre.

Nel secondo, invece, il robot ha un numero di gradi di libertà superiore al numero di componenti del vettore generalizzato di velocità; la matrice jacobiana presenterà alcune colonne linearmente dipendenti e quindi per generare una data velocità nello spazio operativo si hanno diverse possibili combinazioni di velocità nei gradi di libertà dei giunti.

In cinematica inversa, note le componenti del vettore di velocità generalizzata, il vettore delle velocità nei giunti vale:

$$\underline{\dot{q}} = J^{-1} \underline{V}_n$$

Tale relazione sussiste se la matrice jacobiana è invertibile.

Si definiscono a questo proposito “*singolari*” le configurazioni del robot per cui è nullo il determinante della jacobiana: in tal caso la matrice non è invertibile.

Una configurazione singolare implica:

- perdita di libertà di movimento a causa della sovrapposizione dell’effetto di più giunti;
- infinite soluzioni per la cinematica inversa;
- velocità nei giunti che diventano infinite a fronte di una richiesta di movimento lungo la direzione persa a causa della singolarità stessa.

Anche configurazioni prossime alla singolarità presentano problematiche analoghe e sono dunque da evitare in sede di pianificazione della traiettoria.

In questo caso le velocità richieste nei giunti non saranno ancora infinite ma molto grandi, quindi il movimento nella direzione che si sta andando a perdere non sarà precluso ma eseguibile con prestazioni ridotte.

Il robot oggetto della presente relazione è del tipo articolato con polso monocentrico a 3 gradi di libertà, pertanto si distingueranno due possibili tipologie di singolarità: singolarità di braccio e di polso.

Il braccio è l’insieme dei 3 link destinati al posizionamento del punto centro polso, mentre il polso è costituito dai link necessari ad orientare l’*end effector* in modo sferico attorno tale punto.

Il centro polso è il punto discriminante tra braccio e polso.

Per lo studio delle singolarità di braccio si quindi è deciso di modificare la definizione dei sistemi di riferimento di Denavit- Hartenberg adottata in precedenza al fine di portare a coincidere il sistema di riferimento dell’ultimo link con il centro polso (allegato 3).

$$\underline{V}_E = \begin{Bmatrix} \underline{v}_E \\ \underline{\omega}_E \end{Bmatrix} = J_E \begin{Bmatrix} \dot{\vartheta}_1 \\ \dot{\vartheta}_2 \\ \dot{\vartheta}_3 \\ \dot{\vartheta}_4 \\ \dot{\vartheta}_5 \\ \dot{\vartheta}_6 \end{Bmatrix} = \left[ \begin{array}{c|c} J_{11} & 0 \\ \hline J_{21} & J_{22} \end{array} \right] \begin{Bmatrix} \dot{\vartheta}_1 \\ \dot{\vartheta}_2 \\ \dot{\vartheta}_3 \\ \dot{\vartheta}_4 \\ \dot{\vartheta}_5 \\ \dot{\vartheta}_6 \end{Bmatrix}$$

$J_{11}$  = minore 3x3 che rappresenta l'influenza che hanno i gradi di libertà di braccio sulle prime tre componenti della velocità generalizzata

$J_{12}$  = minore 3x3 nullo (in quanto i gradi di libertà di polso non hanno influenza sulla velocità di traslazione dell'*end effector*)

$J_{21}$  = minore 3x3 che rappresenta l'influenza che hanno i gradi di libertà di braccio sulle ultime tre componenti della velocità generalizzata

$J_{22}$  = minore 3x3 che rappresenta l'influenza che hanno i tre gradi di libertà di polso sulle ultime tre componenti della velocità generalizzata

Le condizioni di singolarità:

$$\det(J_E) = \det(J_{11}) \cdot \det(J_{22}) = 0$$

La **singolarità di braccio** si verifica quando:

$$\det(J_{11}) = 0$$

In particolare se:

- due colonne di  $J_{11}$  sono proporzionali tra loro: ad esempio se la seconda e la terza colonna sono proporzionali tra loro nella configurazione in cui il punto centro polso è contenuto nel piano definito dagli assi  $z_2$  e  $z_3$  (figura 30), ne deriva che l'effetto dei giunti 2 e 3 è sovrapposto;

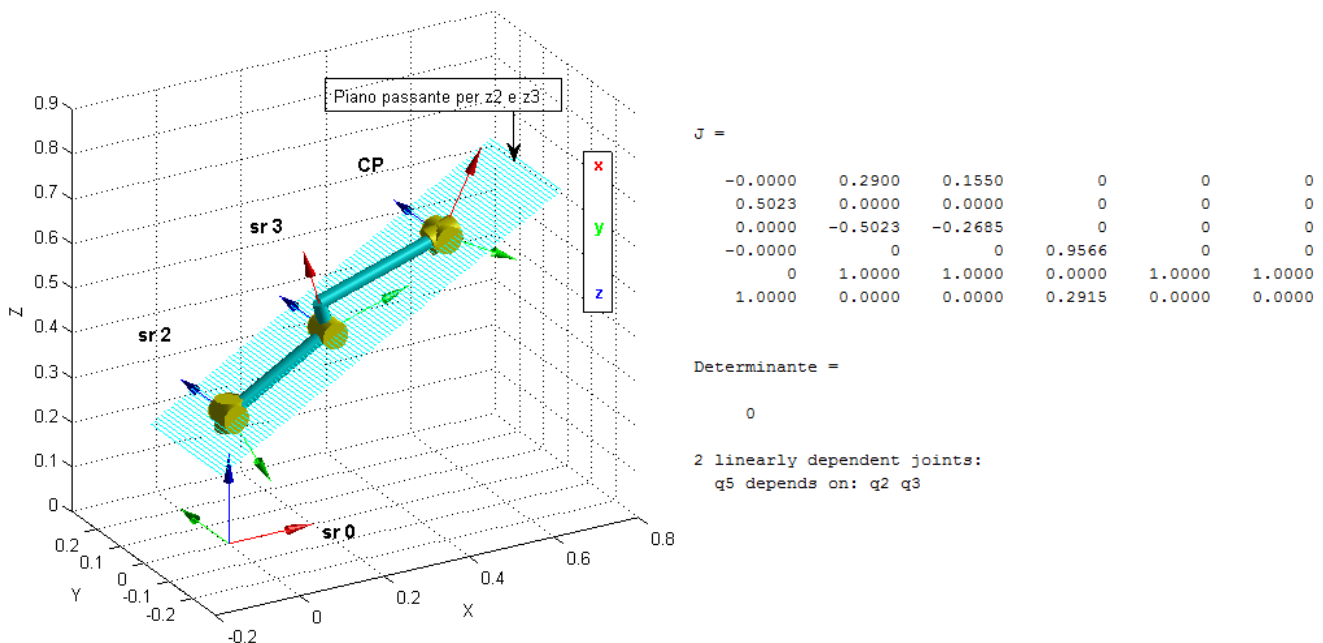


Figura 30: Singolarità di braccio: sovrapposizione effetto  $q_2$  e  $q_3$

- una colonna di  $J_{11}$  è nulla: ad esempio se il centro polso si trova sull'asse  $z_1$  si annulla la prima colonna, il giunto uno potrebbe avere una qualunque velocità nella soluzione della cinematica inversa in questa configurazione, infatti il suo movimento non ha effetti sulle componenti di traslazione del vettore di velocità generalizzata.

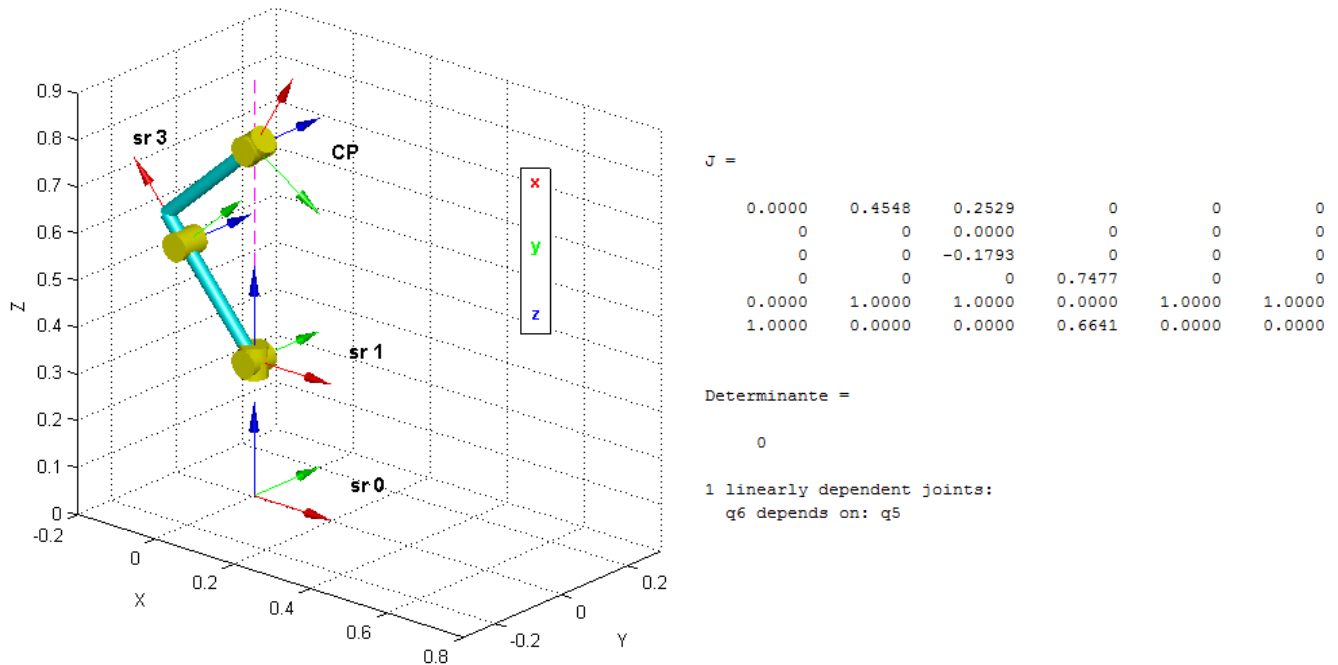


Figura 31: Singolarità di braccio: perdita contributo di  $q_1$

Per lo studio della **singolarità di polso**, i sistemi di riferimento adottati sono nuovamente quelli scelti inizialmente secondo la convenzione di Denavit-Hartenberg.

La singolarità di polso si verifica quando:

$$\det(J_{22}) = 0$$

Tale condizione si verifica se gli assi  $z_4$ ,  $z_5$  e  $z_6$  sono complanari, ovvero quando il grado di libertà nel giunto 5 è nullo oppure uguale a  $180^\circ$ ; ciò comporta la perdita di un grado di libertà in quanto i contributi di  $q_4$  e  $q_6$  si sovrappongono. Inoltre si avranno infinite soluzioni della cinematica inversa e non si potranno generare velocità angolari che non giacciono nel piano  $z_4 z_5$ .

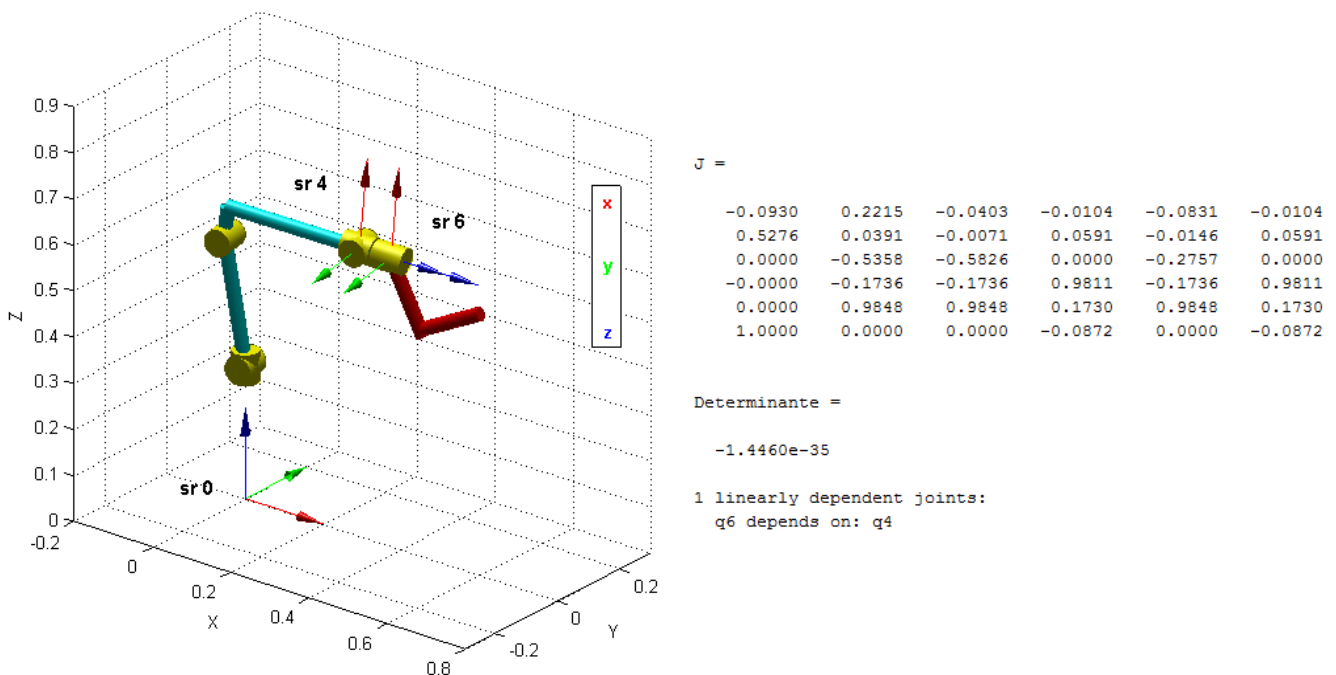


Figura 32: Singolarità di polso: sovrapposizione effetto  $q_4$  e  $q_6$

Come si può notare dalla matrice jacobiana, le colonne 4 e 6 sono uguali: l'influenza dei gradi di libertà 4 e 6 è esattamente la stessa su tutte le componenti del vettore di velocità generalizzata.

## ALLEGATO 1

## Robotics

## IRB 120

## Industrial Robot

## ABB's smallest robot – for flexible and compact production

The IRB 120 robot is the latest addition to ABB's new fourth-generation of robotic technology and ABB's smallest robot ever produced. Ideal for material handling and assembly applications, the new IRB 120 robot provides an agile, compact and lightweight solution with superior control and path accuracy.

**Compact and lightweight**

As the smallest robot from ABB, the IRB 120 offers all the functionality and expertise of the ABB range in a much smaller package. Its reduced weight of only 25kg and compact design enables it to be mounted virtually anywhere, whether it is inside a cell, on top of a machine or close to other robots on the production line.

**Multipurpose**

Ideal for a wide range of industries including the electronic, food and beverage, machinery, solar, pharmaceutical, medical and research sectors, the IRB 120 joins ABB's fourth-generation of new robotic technology.

A white finish Clean Room ISO class 5 version enhances this versatility by making it suitable for environment with stringent cleanliness standard.

The six-axis robot handles a payload of up to 3kg (4kg with its wrist down) and with a reach of 580 mm, the robot is able to carry out a series of operations using flexible rather than hard automated solutions. The IRB 120 is the perfect building block to design cost effective applications – especially when space is at a premium.

**Easy to integrate**

Weighing in at only 25kg, this robot arm is truly the most portable and easy to integrate on the market. It can be mounted at any angle without any restriction. The smooth surfaces are easy to clean and the cables for air and customer signals are internally routed, all the way from the foot to the wrist, ensuring that integration is effortless.

**Optimized working range**

In addition to a horizontal reach of 580 mm, the robot has best in class stroke and the ability to reach 112 mm below its base. Furthermore, the IRB 120 has a very compact turning radius, which is enabled by the robots symmetric architecture, without offset on axis 2. This ensures the robot can be mounted close to other equipment and the slim wrist enables the arm to reach closer to its application.

**Fast, accurate and agile**

Designed with a light, aluminum structure, the powerful compact motors ensure the robot is enabled with a fast acceleration, and can deliver accuracy and agility in any application. Using the IRB 120T variant, cycle-times can be reduced up to 25% where the work piece needs extensive re-orientation and axis 4, 5 and 6 are predominantly used. This faster version is well suited for pick and packing applications and guided operations together with PickMaster 3™.

**IRC5 Compact controller – optimised for small robots**

ABB's new IRC5 Compact controller takes the capabilities of the extremely powerful IRC5 controller and presents them in a truly compact format. The new Compact controller brings accuracy and motion control to applications, which previously had been exclusive to large installations.

In addition to space saving benefits, the new controller also enables easy commissioning through one phase power input, external connectors for all signals and a built-in expandable 16 in, 16 out, I/O system.

RobotStudio for offline programming enables manufacturers to simulate a production cell to find the optimal position for the robot, and provide offline programming to prevent costly downtime and delays to production.

**Reduced footprint**

For applications where floor space is crucial, the combination of the new compact, lightweight architecture of the IRB 120 with the new IRC5 Compact controller introduces a significantly reduced footprint.



## IRB 120

Specification			
Variants	Reach	Payload	Armload
IRB 120-3/0.6	580 mm	3 kg (4kg)*	0.3 kg

### Features

Integrated signal supply	10 signals on wrist
Integrated air supply	4 air on wrist (5 bar)
Position repeatability	0.01 mm
Robot mounting	Any angle
Degree of protection	IP30
Controllers	IRC5 Compact / IRC5 Single cabinet

### Movement

Axis movements	Working range	Maximum speed	
		IRB 120	IRB 120T
Axis 1 Rotation	+165° to -165°	250 °/s	250 °/s
Axis 2 Arm	+110° to -110°	250 °/s	250 °/s
Axis 3 Arm	+70° to -110°	250 °/s	250 °/s
Axis 4 Wrist	+180° to -180°	320 °/s	420 °/s
Axis 5 Bend	+120° to -120°	320 °/s	590 °/s
Axis 6 Turn	+400° to -400°	420 °/s	600 °/s

### Performance

	IRB 120	IRB 120T
<b>1 kg picking cycle</b>		
25 x 300 x 25 mm	0.58 s	0.52 s
25 x 300 x 25 with	0.92 s	0.69 s
180° axis 6 reorientation		
Acceleration time 0-1 m/s	0.07 s	0.07 s

### Electrical connections

Supply voltage	200-600 V, 50/60 Hz
Rated power	
Transformer rating	3.0 kVA
Power consumption	0.25 kW

### Physical

Dimension robot base	180 x 180 mm
Dimension robot height	700 mm
Weight	25 kg

### Environment

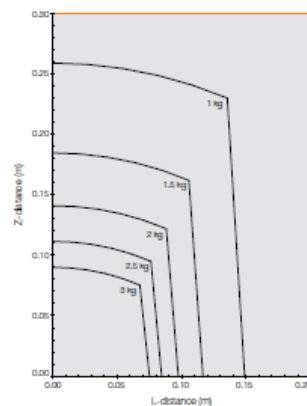
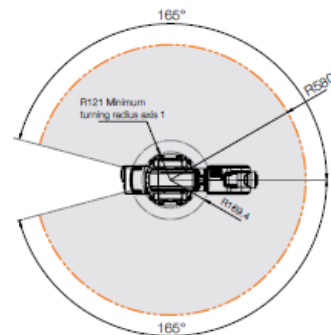
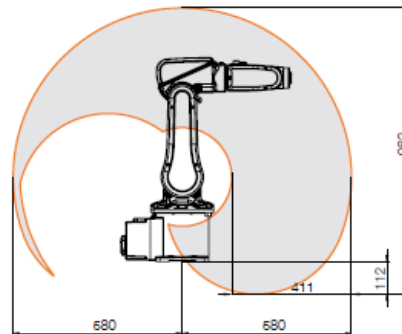
<b>Ambient temperature for Robot manipulator:</b>	
During operation	+5°C (41°F) to +45°C (122°F)
Relative transportation and storage	-25°C (-13°F) to +55°C (131°F)
For short periods	up to +70°C (158°F)
Relative humidity	Max 95%
Options	Clean Room ISO class 5 (certified by IPA)**
Noise level	Max 70 dB (A)
Safety	Safety and emergency stops 2-channel safety circuits supervision 3-position enabling device
Emission	EMC/EMI-shielded

\* With vertical wrist

\*\* ISO class 4 can be reached under certain conditions

Data and dimensions may be changed without notice

### Working range at wrist center & load diagram



## ALLEGATO 2

```
% ROBOT_ABB_IRB_120

clear all
close all
clc

startup_rvc

%% Parametri geometrici (Denavit-Hartenberg)

% Link 1
alpha1=0;
a1=0;
d1=0.290;
teta10=0;
qlim1=[-165 165]*pi/180;    %limiti gdl giunti (da catalogo)

% Link 2
alpha2=-0.5*pi;
a2=0;
d2=0;
teta20=-0.5*pi;
qlim2=[-110 110]*pi/180;    %limiti gdl giunti (da catalogo)

% Link 3
alpha3=0;
a3=0.270;
d3=0;
teta30=0;
qlim3=[-110 70]*pi/180;    %limiti gdl giunti (da catalogo)

% Link 4
alpha4=-0.5*pi;
a4=0.070;
d4=0.302;
teta40=0;
qlim4=[-160 160]*pi/180;    %limiti gdl giunti (da catalogo)

% Link 5
alpha5=0.5*pi;
a5=0;
d5=0;
teta50=0;
qlim5=[-120 120]*pi/180;    %limiti gdl giunti (da catalogo)

% Link 6
alpha6=-0.5*pi;
a6=0;
d6=0.072;
teta60=0;
qlim6=[-400 400]*pi/180;    %limiti gdl giunti (da catalogo)

% descrivo due ulteriori link che userò per analisi dinamica
% (per poter definire le proprietà inerziali di base e tool)

% Link B
```

```

alphaB=0*pi;
aB=0;
dB=0;
tetaB0=0;
qlimB=[0 0]*pi/180; % limiti gdl giunti (non voglio che questi ulteriori
                    % link abbiano un moto relativo rispetto ai precedenti)

% Link T
alphaT=0*pi;
aT=0;
dT=0;
tetaT0=0;
qlimT=[0 0]*pi/180; % limiti gdl giunti (non voglio che questi ulteriori
                    % link abbiano un moto relativo rispetto ai precedenti)

%definizione parametri di D-H
L(1)=Link('alpha',alpha1,'a',a1,'d',d1,'offset',teta10,'modified','qlim',qlim1);
% robot link 1
L(2)=Link('alpha',alpha2,'a',a2,'d',d2,'offset',teta20,'modified','qlim',qlim2);
% robot link 2
L(3)=Link('alpha',alpha3,'a',a3,'d',d3,'offset',teta30,'modified','qlim',qlim3);
% robot link 3
L(4)=Link('alpha',alpha4,'a',a4,'d',d4,'offset',teta40,'modified','qlim',qlim4);
% robot link 4
L(5)=Link('alpha',alpha5,'a',a5,'d',d5,'offset',teta50,'modified','qlim',qlim5);
% robot link 5
L(6)=Link('alpha',alpha6,'a',a6,'d',d6,'offset',teta60,'modified','qlim',qlim6);
% robot link 6

%costruzione robot
ABBirb120=SerialLink(L,'name','ABBirb120'); % to build up a SerialLink
object
%ABBirb120.display

% Assegnazione del vettore dei gradi di libertà

%qvec0=[0 0 0 0 0 0]*pi/180; %vettore dei gradi di libertà dei giunti assegnato
%qvec0=[30 60 45 -50 -80 -20]*pi/180; %vettore dei gradi di libertà dei giunti
assegnato
%qvec0=[20 30 10 -30 30 -20]*pi/180; %vettore dei gradi di libertà dei giunti
assegnato
qvec0=[0 90 -90 0 0 0]*pi/180; %vettore dei gradi di libertà dei giunti
assegnato

zeroAee=ABBirb120.fkine(qvec0) % .fkine calcola la matrice omogenea dell'EE
rispetto al sistema zero (0Aee)

% plot del robot prima di aver assegnato il tool
% figure(1)
% ABBirb120.plot(qvec0) % plot del robot con i gradi di libertà assegnati
% hold on
% trplot(ABBirb120.base,'frame','A0')

% Forza di reazione dovuta al flusso di fluido uscente dall'ugello
% (intensità costante)
forza=30; %[N]

% Accelerazione gravitazionale
%g=9.80665; %[m/s^2]
g=9.81; % [m/s^2]

```

```

% Parametri tool ugello di verniciatura
angolo=60*pi/180;
ics=-0.060;
zeta=0.210;

% Matrice eeAtcp
eeAtcp=[cos(angolo)  0  -sin(angolo)  ics
        0            1   0           0
        sin(angolo)  0   cos(angolo)  zeta
        0            0   0           1   ];

ABBBirb120.tool=eeAtcp;
ABBBirb120.display

% plot del robot dopo di aver assegnato il tool
figure(1)
ABBBirb120.plot(qvec0) % plot del robot con i gradi di libertà assegnati
hold on
trplot(ABBBirb120.base,'frame','A0')

%% PLOT PER I SISTEMI DI RIFERIMENTO DI D-H

qvec=[0 0 0 0 0 0];
[zeroAtcp, A_all]=ABBBirb120.fkine(qvec);
zeroA1=A_all(:, :, 1);
zeroA2=A_all(:, :, 2);
zeroA3=A_all(:, :, 3);
zeroA4=A_all(:, :, 4);
zeroA5=A_all(:, :, 5);
zeroA6=A_all(:, :, 6);
zeroAtcp;

% Plot con i sistemi di riferimento: base,1,2,5,6,tool
figure
ABBBirb120.plot([qvec], 'workspace', [-0.2, 0.8, -
0.3, 0.3, 0, 0.9], 'floorlevel', 0, 'notiles', 'noname', 'nobase', 'noshadow', 'jointdiam'
, 4.5, 'jointcolor', 'y', 'zoom', 1, 'nowrist', 'linkcolor', 'c')
hold on
trplot(ABBBirb120.base, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA1, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA3, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA5, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA6, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroAtcp, 'notext', 'length', 0.2, 'rgb', 'arrow')
legend('x', 'y', 'z')

% Plot con i sistemi di riferimento: base,2,4,tool
figure
ABBBirb120.plot([qvec], 'workspace', [-0.2, 0.8, -
0.3, 0.3, 0, 0.9], 'floorlevel', 0, 'notiles', 'noname', 'nobase', 'noshadow', 'jointdiam'
, 4.5, 'jointcolor', 'y', 'zoom', 1, 'nowrist', 'linkcolor', 'c')
hold on
trplot(ABBBirb120.base, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA2, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA4, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroAtcp, 'notext', 'length', 0.2, 'rgb', 'arrow')

%% ANALISI SINGOLARITA'

```

```

%% SINGOLARITA' BRACCIO CONFIGURAZIONE Q2 E Q3

u=0.07;
f=0.302;
w=atan(u/f);
qvec=[0 pi/3 -pi/2+w 0 pi/4 pi/4];

%% SINGOLARITA' BRACCIO D3=0 Q1 ININFLUENTE

u=0.302;
f=0.270-0.070;
w=atan(70/302);
h=acos(270/sqrt(302^2+70^2))*sin(pi/3);
qvec=[0 -atan(302/(270+70)) 0 0 pi/2+atan(302/340) 0];

%% SINGOLARTIA' POLSO

qvec=pi/180*[10 -10 15 0 0 0];

%% Plot delle singolarità

[zeroAtcp, A_all]=ABBirb120.fkine(qvec);
zeroA1=A_all(:, :, 1);
zeroA2=A_all(:, :, 2);
zeroA3=A_all(:, :, 3);
zeroA4=A_all(:, :, 4);
zeroA5=A_all(:, :, 5);
zeroA6=A_all(:, :, 6);
zeroAtcp;
J= ABBirb120.jacob0(qvec)
Determinante=det(J)
jsingu(J)

figure
ABBirb120.plot([qvec], 'workspace', [-0.2, 0.8, -
0.3, 0.3, 0, 0.9], 'floorlevel', 0, 'notiles', 'noname', 'nobase', 'noshadow', 'jointdiam'
, 4.5, 'jointcolor', 'y', 'zoom', 1, 'nowrist', 'linkcolor', 'c')
hold on
trplot(ABBirb120.base, 'notext', 'length', 0.2, 'rgb', 'arrow')
% trplot(zeroA1, 'notext', 'length', 0.2, 'rgb', 'arrow')
% trplot(zeroA3, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA4, 'notext', 'length', 0.2, 'rgb', 'arrow')
% trplot(zeroA5, 'notext', 'length', 0.2, 'rgb', 'arrow')
trplot(zeroA6, 'notext', 'length', 0.2, 'rgb', 'arrow')
% trplot(zeroAtcp, 'notext', 'length', 0.2, 'rgb', 'arrow')
legend('x', 'y', 'z')

%% Matrici di tutti i corpi rispetto al sistema zero
[zeroAtcp, A_all]=ABBirb120.fkine(qvec0);

zeroA0=eye(4);

zeroA1=A_all(:, :, 1);
zeroA2=A_all(:, :, 2);
zeroA3=A_all(:, :, 3);
zeroA4=A_all(:, :, 4);
zeroA5=A_all(:, :, 5);

```

```

zeroA6=A_all(:, :, 6);
zeroAtcp;

% CALCOLI PER PLOTTAGGIO CAD DEL ROBOT CON SIMULINK

% Matrici S.R. CAD wrt S.R. D-H

dh0Acad0=[1  0  0  0
           0  1  0  0
           0  0  1  0
           0  0  0  1];

dh1Acad1=[1  0  0  0
           0  1  0  0
           0  0  1  0
           0  0  0  1];

dh2Acad2=[0  0  1  a3
           1  0  0  0
           0  1  0  0
           0  0  0  1 ];

dh3Acad3=[0  0  1  0
           1  0  0  0
           0  1  0  0
           0  0  0  1];

dh4Acad4=[0  0  1  0
           0 -1  0  0
           1  0  0  0
           0  0  0  1];

dh5Acad5=[0  0  1  0
           1  0  0  0
           0  1  0  0
           0  0  0  1];

dh6Acad6=[0  0  1  0
           0 -1  0  0
           1  0  0  0
           0  0  0  1];

dhTAcadT=dh6Acad6; %la utilizzo più avanti per trasferire le info di baricentro
                  % e tensore di inerzia nel S.R. di D-H

tcpAcadt=inv([sin(angolo)  0  cos(angolo)  zeta
              0            -1  0          0
              cos(angolo)  0 -sin(angolo)  ics
              0            0  0          1  ]);

% MATRICI SR_CAD (SERVONO PER IL PLOTTAGGIO DEL ROBOT CON SIMULINK)

zeroAcad0=zeroA0*dh0Acad0;
zeroAcad1=zeroA1*dh1Acad1;
zeroAcad2=zeroA2*dh2Acad2;
zeroAcad3=zeroA3*dh3Acad3;
zeroAcad4=zeroA4*dh4Acad4;
zeroAcad5=zeroA5*dh5Acad5;
zeroAcad6=zeroA6*dh6Acad6;

```

```

zeroAcadt=zeroAtcp*tcpAcadt;

%% PLOT CAD DEL ROBOT CON MECHANICS EXPLORER

plot_IRB120(zeroAcad1,zeroAcad2,zeroAcad3,zeroAcad4,zeroAcad5,zeroAcad6)
%plot CAD senza tool
plot_IRB120tool(zeroAcad1,zeroAcad2,zeroAcad3,zeroAcad4,zeroAcad5,zeroAcad6,zero
Acadt) %plot CAD con tool
delete *.asv % Cleaning

%% Proprietà inerziali

% Masse
m0=9.62116; %[kg]
m1=4.25439; %[kg]
m2=4.42193; %[kg]
m3=4.58341; %[kg]
m4=1.34194; %[kg]
m5=0.75825; %[kg]
m6=0.01892; %[kg]
mT=0.43649; %[kg]

% Posizioni origini S.R. Centri di massa (baricentri) wrt S.R. CAD
CM0=[-42.04,0.08,79.64]*1e-3; %[m]
CM1=[0.10,-0.12,-51.59]*1e-3; %[m]
CM2=[0.78,-2.12,-168.76]*1e-3; %[m]
CM3=[22.81,1.06,57.91]*1e-3; %[m]
CM4=[-77.30,0.15,0.41]*1e-3; %[m]
CM5=[-1.09,0.04,0.06]*1e-3; %[m]
CM6=[-7.06,-0.17,0.00]*1e-3; %[m]
CMT=[98.57,0.00,11.37]*1e-3; %[m]

CM0o=[CM0,1];
CM1o=[CM1,1];
CM2o=[CM2,1];
CM3o=[CM3,1];
CM4o=[CM4,1];
CM5o=[CM5,1];
CM6o=[CM6,1];
CMTo=[CMT,1];

% Matrici orientazione S.R. Centri di massa wrt S.R. CAD
cad0Acm0=[ 0.9983, 0.0581, 0.0000
           0.0000, 0.0000, 1.0000
           0.0581,-0.9983, 0.0000];

cad1Acm1=[-0.0062, 0.9977, 0.0678
           0.0049,-0.0678, 0.9977
           0.9999, 0.0065,-0.0045];

cad2Acm2=[ 0.0000, 0.0000, 1.0000
           -0.0323,-0.9994, 0.0000
           0.9995,-0.0322, 0.0000];

cad3Acm3=[ 0.9578, 0.2876, 0.0020
           -0.0147, 0.0421, 0.9990
           0.2872,-0.9568, 0.0446];

```

```

cad4Acm4=[ 0.9998, 0.0184, 0.0067
           -0.0183, 0.9998,-0.0105
           -0.0069, 0.0103, 0.9999];

cad5Acm5=[ 1.0000, 0.0000, 0.0000
           0.0000, 0.0000, 1.0000
           0.0000,-1.0000, 0.0000];

cad6Acm6=[ 0.0000, 0.0000, 1.0000
           0.0000,-1.0000, 0.0000
           1.0000, 0.0000, 0.0000];

cadTAcmT=[-0.9826, 0.1857, 0.0000
           0.0000, 0.0000,-1.0000
           -0.1857,-0.9826, 0.0000];

% Tensori centrali di inerzia (wrt S.R. CM che è un S.R. centrale
% (=principale+baricentrico))
IG0=[38161020.67, 0.000000000, 0.000000000
     0.000000000, 73243624.28, 0.000000000
     0.000000000, 0.000000000, 76053785.67]*1e-9; %[kg*m^2]

IG1=[14500004.44, 0.000000000, 0.000000000
     0.000000000, 19720761.52, 0.000000000
     0.000000000, 0.000000000, 19982083.09]*1e-9; %[kg*m^2]

IG2=[29344594.91, 0.000000000, 0.000000000
     0.000000000, 47046106.12, 0.000000000
     0.000000000, 0.000000000, 68231040.33]*1e-9; %[kg*m^2]

IG3=[12341191.27, 0.000000000, 0.000000000
     0.000000000, 20428062.83, 0.000000000
     0.000000000, 0.000000000, 26035241.49]*1e-9; %[kg*m^2]

IG4=[2876276.090, 0.000000000, 0.000000000
     0.000000000, 4056258.710, 0.000000000
     0.000000000, 0.000000000, 5309447.470]*1e-9; %[kg*m^2]

IG5=[561630.6400, 0.000000000, 0.000000000
     0.000000000, 1131171.140, 0.000000000
     0.000000000, 0.000000000, 1238480.600]*1e-9; %[kg*m^2]

IG6=[2292.670000, 0.000000000, 0.000000000
     0.000000000, 2343.180000, 0.000000000
     0.000000000, 0.000000000, 4117.120000]*1e-9; %[kg*m^2]

IGT=[179916.3000, 0.000000000, 0.000000000
     0.000000000, 1572293.230, 0.000000000
     0.000000000, 0.000000000, 1674963.520]*1e-9; %[kg*m^2]

% Posizioni origini S.R. Centri di massa wrt S.R. D.H.

b0o=dh0Acad0*CM0o';
b1o=dh1Acad1*CM1o';
b2o=dh2Acad2*CM2o';
b3o=dh3Acad3*CM3o';
b4o=dh4Acad4*CM4o';
b5o=dh5Acad5*CM5o';
b6o=dh6Acad6*CM6o';

```



```

bTo=dhTAcadT*CMT0';

b0=b0o(1:3,1);
b1=b1o(1:3,1);
b2=b2o(1:3,1);
b3=b3o(1:3,1);
b4=b4o(1:3,1);
b5=b5o(1:3,1);
b6=b6o(1:3,1);
bT=bTo(1:3,1);

% Tensori di inerzia (wrt S.R. D.H. ma baricentrico)

dhIG0=(dh0Acad0(1:3,1:3)*cad0Acm0)*IG0*((dh0Acad0(1:3,1:3)*cad0Acm0)');
dhIG1=(dh1Acad1(1:3,1:3)*cad1Acm1)*IG1*((dh1Acad1(1:3,1:3)*cad1Acm1)');
dhIG2=(dh2Acad2(1:3,1:3)*cad2Acm2)*IG2*((dh2Acad2(1:3,1:3)*cad2Acm2)');
dhIG3=(dh3Acad3(1:3,1:3)*cad3Acm3)*IG3*((dh3Acad3(1:3,1:3)*cad3Acm3)');
dhIG4=(dh4Acad4(1:3,1:3)*cad4Acm4)*IG4*((dh4Acad4(1:3,1:3)*cad4Acm4)');
dhIG5=(dh5Acad5(1:3,1:3)*cad5Acm5)*IG5*((dh5Acad5(1:3,1:3)*cad5Acm5)');
dhIG6=(dh6Acad6(1:3,1:3)*cad6Acm6)*IG6*((dh6Acad6(1:3,1:3)*cad6Acm6)');
dhIGT=(dhTAcadT(1:3,1:3)*cadTAcmT)*IGT*((dhTAcadT(1:3,1:3)*cadTAcmT)');

% PROPRIETA' INERZIALI DEI LINK GIA' ESISTENTI

L(1).m=m1;
L(1).r=b1;
L(1).I=dhIG1;

L(2).m=m2;
L(2).r=b2;
L(2).I=dhIG2;

L(3).m=m3;
L(3).r=b3;
L(3).I=dhIG3;

L(4).m=m4;
L(4).r=b4;
L(4).I=dhIG4;

L(5).m=m5;
L(5).r=b5;
L(5).I=dhIG5;

L(6).m=m6;
L(6).r=b6;
L(6).I=dhIG6;

% NUOVA COSTRUZIONE DEL ROBOT
%(AGGIUNGO LA BASE E IL TOOL PER POTERNE DEFINIRE LE PROPRIETA' INERZIALI)

%definizione parametri di D-H

T(1)=Link('alpha',alphaT,'a',aT,'d',dT,'offset',tetaT0,'modified','qlim',qlimT);
%robot link tool
tool=SerialLink(T,'name','tool'); % to build up a SerialLink object

T(1).m = mT;
T(1).r = bT;

```

```

T(1).I = dhIGT;

B(1)=Link('alpha',alphaB,'a',aB,'d',dB,'offset',tetaB0,'modified','qlim',qlimB);
%robot link base
base=SerialLink(B,'name','base'); % to build up a SerialLink object

B(1).m = m0;
B(1).r = b0;
B(1).I = dhIG0;

ABBBirb120_Dyn=SerialLink([base,ABBBirb120,tool],'name','ABBBirb120_D_y_n');
ABBBirb120_Dyn.tool=eeAtcp;

figure(2)
ABBBirb120_Dyn
ABBBirb120_Dyn.plot([0,qvec0,0])
hold on
trplot(ABBBirb120_Dyn.base,'frame','A0')

%% Mappatura spazio di lavoro
q2vec=linspace(min(qlim2),max(qlim2),50);
q3vec=linspace(min(qlim3),max(qlim3),50);

i23=1;
for i2=1:length(q2vec)
    for i3=1:length(q3vec)
        q1=0;
        q2=q2vec(i2);
        q3=q3vec(i3);
        q4=0;
        q5=0;
        q6=0;
        zeroA4=ABBBirb120.A([1 2 3 4],[q1 q2 q3 q4]); % Centro Polso
        zeroA6=ABBBirb120.A([1 2 3 4 5 6],[q1 q2 q3 q4 q5 q6]); % End Effector
        C4=zeroA4(1:3,4);
        C6=zeroA6(1:3,4);
        wspace(i23,:)= [q1 q2 q3 q4 q5 q6 C4' C6' 0];
        i23=i23+1;
    end
end

% figure(2)
% plot(wspace(:,7),wspace(:,9),'*')
% axis equal

% nel ciclo for cerco i punti da escludere dallo spazio di lavoro potenziale del
robot

for iCR=1:numrows(wspace)
    if wspace(iCR,9)<0 | wspace(iCR,9)<0.2 & abs(wspace(iCR,7))<0.1
        wspace(iCR,13)=1; % flag posto a uno per i valori critici
    end
end

wspace01=sortrows(wspace,13); % ordino le righe di wspace in modo che si abbiano
prima quelle con flag=0 e poi quelle con flag=1
row1=find(wspace01(:,13),1); % trovo la prima riga in cui si ha flag=1 e ne
salvo l'indice in row1

```

```

wspace0=wspace01(1:row1-1,:); % porzione del wspace contenente i punti
raggiungibili
wspace1=wspace01(row1:numrows(wspace01),:); % porzione del wspace contenente i
punti critici

%wspace CP
figure(3)
subplot(2,2,1)
plot(radtodeg(wspace(:,2)),radtodeg(wspace(:,3)),'*')
axis equal
xlabel('q2 [deg]')
ylabel('q3 [deg]')
title('spazio q3 vs q2')

subplot(2,2,2)
plot(wspace(:,7),wspace(:,9),'*')
axis equal
xlabel('x')
ylabel('z')
title('wspace CP')

subplot(2,2,3)
plot(radtodeg(wspace0(:,2)),radtodeg(wspace0(:,3)),'b*')
hold on
plot(radtodeg(wspace1(:,2)),radtodeg(wspace1(:,3)),'r*')
axis equal
xlabel('q2 [deg]')
ylabel('q3 [deg]')
title('spazio q3 vs q2 punti critici in rosso');

subplot(2,2,4)
plot(wspace0(:,7),wspace0(:,9),'b*')
hold on
plot(wspace1(:,7),wspace1(:,9),'r*')
axis equal
xlabel('x')
ylabel('z')
title('wspace CP punti critici in rosso');

%% TRAIETTORIA CIRCOLARE

load('rvctools\traiettoria_6gdl_v02.mat');

% PLOTTAGGIO DEI GDL PER OGNI GIUNTO

figure(4)
subplot(3,2,1)
plot(tempovet,q1vet,tempovet,q1pvet,tempovet,q1ppvet),grid on
xlabel('Tempo (s)')
legend('q1[deg]','q1p[deg/s]','q1pp[deg/s^2]')
title('gdl Giunto 1')

subplot(3,2,2)
plot(tempovet,q2vet,tempovet,q2pvet,tempovet,q2ppvet),grid on
xlabel('Tempo (s)')
legend('q2[deg]','q2p[deg/s]','q2pp[deg/s^2]')
title('gdl Giunto 2')

```

```

subplot(3,2,3)
plot(tempovet,q3vet,tempovet,q3pvet,tempovet,q3ppvet),grid on
xlabel('Tempo (s)')
legend('q3[deg]','q3p[deg/s]','q3pp[deg/s^2]')
title('gdl Giunto 3')

subplot(3,2,4)
plot(tempovet,q4vet,tempovet,q4pvet,tempovet,q4ppvet),grid on
xlabel('Tempo (s)')
legend('q4[deg]','q4p[deg/s]','q4pp[deg/s^2]')
title('gdl Giunto 4')

subplot(3,2,5)
plot(tempovet,q5vet,tempovet,q5pvet,tempovet,q5ppvet),grid on
xlabel('Tempo (s)')
legend('q5[deg]','q5p[deg/s]','q5pp[deg/s^2]')
title('gdl Giunto 5')

subplot(3,2,6)
plot(tempovet,q6vet,tempovet,q6pvet,tempovet,q6ppvet),grid on
xlabel('Tempo (s)')
legend('q6[deg]','q6p[deg/s]','q6pp[deg/s^2]')
title('gdl Giunto 6')

% calcolo del vettore di velocità generalizzata per ogni tempo i
for i=1:length(tempovet)

    qvec_i=deg2rad([q1vet(i),q2vet(i),q3vet(i),q4vet(i),q5vet(i),q6vet(i)]);
    qpvec_i=deg2rad([q1pvet(i),q2pvet(i),q3pvet(i),q4pvet(i),q5pvet(i),q6pvet(i)]);

    J(:, :, i)=ABBirb120.jacob0(qvec_i);
    V(:, i)=J(:, :, i)*qpvec_i';

    [zeroAtcp, A_all]=ABBirb120.fkine(qvec_i);

    zeropCP(:, i)=A_all(1:3,4,4);
    zeropEE(:, i)=A_all(1:3,4,6);
    zeropTcp(:, i)=zeroAtcp(1:3,4);

end

% jsingu(J(:, :, i))

% PLOTTAGGIO VETTORE DI VELOCITA' GENERALIZZATA

figure(5)

subplot(2,1,1)
plot(tempovet,V(1,:),tempovet,V(2,:),tempovet,V(3,:),tempovet,sqrt(V(1,:).^2+V(2, :).^2+V(3,:).^2)),grid on
xlabel('tempo [s]');
title('Velocità Tool Centre Point v_T')
legend('v_T_x [m/s]','v_T_y [m/s]','v_T_z [m/s]','|v_T| [m/s]')

subplot(2,1,2)
plot(tempovet,V(4,:),tempovet,V(5,:),tempovet,V(6,:)),grid on
xlabel('tempo [s]');

```

```

title('Velocità angolare Tool w_T')
legend('w_T_x [rad/s]', 'w_T_y [rad/s]', 'w_T_z [rad/s]')

% PLOTTAGGIO TRAIETTORIA E PERCORSO CP

figure(6)

subplot(2,2,1)
plot(tempovet, zeropCP(1,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('x_C_P [m]')

subplot(2,2,2)
plot(tempovet, zeropCP(2,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('y_C_P [m]')

subplot(2,2,3)
plot(tempovet, zeropCP(3,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('z_C_P [m]')

subplot(2,2,4)
plot3(zeropCP(1,:), zeropCP(2,:), zeropCP(3,:), '-c', 'Linewidth', 2), grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso CP')

% PLOTTAGGIO TRAIETTORIA E PERCORSO EE

figure(7)

subplot(2,2,1)
plot(tempovet, zeropEE(1,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('x_E_E [m]')

subplot(2,2,2)
plot(tempovet, zeropEE(2,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('y_E_E [m]')

subplot(2,2,3)
plot(tempovet, zeropEE(3,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('z_E_E [m]')

subplot(2,2,4)
plot3(zeropEE(1,:), zeropEE(2,:), zeropEE(3,:), '-m', 'Linewidth', 2), grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso EE')

% PLOTTAGGIO TRAIETTORIA E PERCORSO TCP

figure(8)

```

```

subplot(2,2,1)
plot(tempovet,zeropTcp(1,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('x_T_C_P [m]')

subplot(2,2,2)
plot(tempovet,zeropTcp(2,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('y_T_C_P [m]')

subplot(2,2,3)
plot(tempovet,zeropTcp(3,:), '-b'), grid on
xlabel('tempo [s]')
ylabel('z_T_C_P [m]')

subplot(2,2,4)
plot3(zeropTcp(1,:),zeropTcp(2,:),zeropTcp(3,:), '-y', 'Linewidth',2), grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso TCP')

%ricavo una delle 1001 configurazioni del robot mentre compie la traiettoria
assegnata

%for riga=1:13:1001
riga=850;
qvecP=deg2rad([q1vet(riga),q2vet(riga),q3vet(riga),q4vet(riga),q5vet(riga),q6vet
(riga)]);

figure(9)
ABBBirb120_Dyn.plot([0,qvecP,0], 'notiles', 'zoom',2) % plot del robot con i gradi
di libertà assegnati
hold on
plot3(zeropCP(1,:),zeropCP(2,:),zeropCP(3,:), '-c', 'Linewidth',2)
plot3(zeropEE(1,:),zeropEE(2,:),zeropEE(3,:), '-m', 'Linewidth',2)
plot3(zeropTcp(1,:),zeropTcp(2,:),zeropTcp(3,:), '-y', 'Linewidth',2)
% trplot(ABBBirb120.base, 'frame', 'A0')
%end

%% DINAMICA

q0vet=zeros(1001,1);

Q=deg2rad([q0vet,q1vet,q2vet,q3vet,q4vet,q5vet,q6vet,q0vet]);
QD=deg2rad([q0vet,q1pvet,q2pvet,q3pvet,q4pvet,q5pvet,q6pvet,q0vet]);
QDD=deg2rad([q0vet,q1ppvvet,q2ppvvet,q3ppvvet,q4ppvvet,q5ppvvet,q6ppvvet,q0vet]);

GRAV=[0;0;g];
tcpFe=[0,0,forza]';
tcpMe=[0,0,0]';

tFet=eeAtcp(1:3,1:3)*tcpFe;
tMet=eeAtcp(1:3,1:3)*tcpMe+cross(eeAtcp(1:3,4),eeAtcp(1:3,1:3)*tcpFe);

FEXT=[tFet;tMet];

for h=1:length(tempovet)

```

```
[TAU(h,:),WBASE(:,h)]=ABBirb120_Dyn.rne_mdh_POLITO([Q(h,:),QD(h,:),QDD(h,:)],GRA  
V,FEXT');
```

```
end
```

```
% PLOTTAGGIO DELLE AZIONI MOTRICI
```

```
figure (10)
```

```
subplot(3,2,1)  
plot(tempovet,TAU(:,2)),grid on  
xlabel('tempo [s]')  
ylabel('taul [Nm]')  
title('azione motrice 1')
```

```
subplot(3,2,2)  
plot(tempovet,TAU(:,3)),grid on  
xlabel('tempo [s]')  
ylabel('tau2 [Nm]')  
title('azione motrice 2')
```

```
subplot(3,2,3)  
plot(tempovet,TAU(:,4)),grid on  
xlabel('tempo [s]')  
ylabel('tau3 [Nm]')  
title('azione motrice 3')
```

```
subplot(3,2,4)  
plot(tempovet,TAU(:,5)),grid on  
xlabel('tempo [s]')  
ylabel('tau4 [Nm]')  
title('azione motrice 4')
```

```
subplot(3,2,5)  
plot(tempovet,TAU(:,6)),grid on  
xlabel('tempo [s]')  
ylabel('tau5 [Nm]')  
title('azione motrice 5')
```

```
subplot(3,2,6)  
plot(tempovet,TAU(:,7)),grid on  
xlabel('tempo [s]')  
ylabel('tau6 [Nm]')  
title('azione motrice 6')
```

```
% PLOTTAGGIO DELLE AZIONI TRASMESSE A PAVIMENTO
```

```
figure (11)
```

```
subplot(2,2,1)  
plot(tempovet,-WBASE(1,:),'-r'),grid on  
xlabel('tempo [s]')  
ylabel('F_p_x [N]')
```

```
subplot(2,2,2)  
plot(tempovet,-WBASE(2,:),'-r'),grid on  
xlabel('tempo [s]')  
ylabel('F_p_y [N]')
```

```
subplot(2,2,3)
plot(tempovet,-WBASE(3,:), '-r'), grid on
xlabel('tempo [s]')
ylabel('F_p_z [N]')
```

figure (12)

```
subplot(2,2,1)
plot(tempovet,-WBASE(4,:), '-r'), grid on
xlabel('tempo [s]')
ylabel('M_p_x [Nm]')
```

```
subplot(2,2,2)
plot(tempovet,-WBASE(5,:), '-r'), grid on
xlabel('tempo [s]')
ylabel('M_p_y [Nm]')
```

```
subplot(2,2,3)
plot(tempovet,-WBASE(6,:), '-r'), grid on
xlabel('tempo [s]')
ylabel('M_p_z [Nm]')
```



## ALLEGATO 3

```
% ANALISI SINGOLARITA'

clear all
close all
clc

startup_rvc

%% PARAMETRI GEOMETRICI DENAVIT-HARTENBERG
alfa1 = 0 ; a1 = 0 ; d1 = 290e-3 ;
alfa2 = -pi/2 ; a2 = 0 ; d2 = 0 ;
alfa3 = 0 ; a3 = 270e-3 ; d3 = 0 ;
alfa4 = -pi/2 ; a4 = 70e-3 ; d4 = 302e-3 ;
alfa5 = pi/2 ; a5 = 0 ; d5 = 0 ;
alfa6 = 0 ; a6 = 0 ; d6 = 0 ;

alfaT=0; aT=0; dT=0; tetaT0=0;
alfaB=0; aB=0; dB=0; tetaB0=0;

%Parametri tool
incl=60*pi/180;
a=0.06;
b=0.21;

% Forza di reazione dovuta al flusso di fluido uscente dall'ugello
% (intensità costante)
forza=30; % [N]

%% LIMITI DEI GRADI DI LIBERTA' DEI GIUNTI

qlim1=[-165 165]*pi/180;
qlim2=[-110 110]*pi/180;
qlim3=[-110 70]*pi/180;
qlim4=[-160 160]*pi/180;
qlim5=[-120 120]*pi/180;
qlim6=[-400 400]*pi/180;

%% LINK DEI ROBOT

L(1)=Link('alpha', alfa1, 'a', a1, 'd', d1, 'modified', 'qlim', qlim1);
L(2)=Link('alpha', alfa2, 'a', a2, 'd', d2, 'offset', -pi/2, 'modified', 'qlim', qlim2);
L(3)=Link('alpha', alfa3, 'a', a3, 'd', d3, 'modified', 'qlim', qlim3);
L(4)=Link('alpha', alfa4, 'a', a4, 'd', d4, 'modified', 'qlim', qlim4);
L(5)=Link('alpha', alfa5, 'a', a5, 'd', d5, 'modified', 'qlim', qlim5);
L(6)=Link('alpha', alfa6, 'a', a6, 'd', d6, 'modified', 'qlim', qlim6);

%% COSTRUZIONE DEL ROBOT

RobotIRB120=SerialLink(L, 'name', 'RobotIRB120');

%% SINGOLARITA' BRACCIO CONFIGURAZIONE Q2 E Q3

u=0.07;
```

```

f=0.302;
w=atan(u/f);
qvec=[0 pi/3 -pi/2+w 0 pi/4 0];

%% SINGOLARITA' BRACCIO D3=0 Q1 ININFLUENTE

u=0.302;
f=0.270-0.070;
w=atan(70/302);
h=acos(270/sqrt(302^2+70^2))*sin(pi/3);
qvec=[0 -atan(302/(270+70)) 0 0 pi/2.5 0];

%% PLOT
format loose

[zeroAtcp A_all]=RobotIRB120.fkine(qvec);
zeroA1=A_all(:, :, 1);
zeroA2=A_all(:, :, 2);
zeroA3=A_all(:, :, 3);
zeroA4=A_all(:, :, 4);
zeroA5=A_all(:, :, 5);
zeroA6=A_all(:, :, 6);
zeroAtcp;

J=RobotIRB120.jacob0(qvec)
Determinante=det(J)
jsingu(J)

z=linspace(0.29,0.9,10);
x=zeros(10);

figure
RobotIRB120.plot([qvec], 'workspace', [-0.2,0.8,-
0.3,0.3,0,0.9], 'floorlevel',0, 'notiles', 'noname', 'nobase', 'noshadow', 'jointdiam'
,4.5, 'jointcolor', 'y', 'zoom',1, 'nowrist', 'linkcolor', 'c', 'lightpos', [20 -20
20])
hold on
trplot(RobotIRB120.base, 'notext', 'length',0.2, 'rgb', 'arrow')
trplot(zeroA1, 'notext', 'length',0.2, 'rgb', 'arrow')
% trplot(zeroA2, 'notext', 'length',0.2, 'rgb', 'arrow')
trplot(zeroA3, 'notext', 'length',0.2, 'rgb', 'arrow')
trplot(zeroA5, 'notext', 'length',0.2, 'rgb', 'arrow')
% trplot(zeroA6, 'notext', 'length',0.2, 'rgb', 'arrow')
% trplot(zeroAtcp, 'notext', 'length',0.2, 'rgb', 'arrow')
legend('x', 'y', 'z')
% [X Y] = meshgrid(x,y);
% plot3(X,Y, (0.0135*X-8.5e-6*Y+0.0068)/0.0234, 'c'), grid on

[X Z] = meshgrid(x,z);
plot3(X, 0*X, Z, '--m')

```

**ALLEGATO 4**

```

function [sol6gdl] = cin_inversa(zeroA6)

alfa1 = 0 ; a1 = 0 ; d1 = 290e-3 ;
alfa2 = -pi/2 ; a2 = 0 ; d2 = 0 ;
alfa3 = 0 ; a3 = 270e-3 ; d3 = 0 ;
alfa4 = -pi/2 ; a4 = 70e-3 ; d4 = 302e-3 ;
alfa5 = pi/2 ; a5 = 0 ; d5 = 0 ;
alfa6 = -pi/2 ; a6 = 0 ; d6 = 72e-3 ;

incl=60*pi/180;
a=0.06;
b=0.21;

qlim1=[-165 165]*pi/180;
qlim2=[-110 110]*pi/180;
qlim3=[-110 70]*pi/180;
qlim4=[-160 160]*pi/180;
qlim5=[-120 120]*pi/180;
qlim6=[-400 400]*pi/180;

L(1)=Link('alpha', alfa1, 'a', a1, 'd', d1, 'modified');%, 'qlim', qlim1);
L(2)=Link('alpha', alfa2, 'a', a2, 'd', d2, 'offset', -
pi/2, 'modified');%, 'qlim', qlim2);
L(3)=Link('alpha', alfa3, 'a', a3, 'd', d3, 'modified');%, 'qlim', qlim3);
L(4)=Link('alpha', alfa4, 'a', a4, 'd', d4, 'modified');%, 'qlim', qlim4);
L(5)=Link('alpha', alfa5, 'a', a5, 'd', d5, 'modified');%, 'qlim', qlim5);
L(6)=Link('alpha', alfa6, 'a', a6, 'd', d6, 'modified');%, 'qlim', qlim6);

RobotIRB120=SerialLink(L, 'name', 'RobotIRB120');

seiAtcp=[cos(incl) 0 -sin(incl) -a;
         0 1 0 0;
         sin(incl) 0 cos(incl) b;
         0 0 0 1];
RobotIRB120.tool=seiAtcp;

% preallocazione matrici soluzioni braccio 3gdl e robot 6gdl completo
solbr=NaN(4,3);
sol6gdl=NaN(8,6);

% calcolo posizione centro polso nota matrice sr6 wrt sr0
pCP0=zeroA6(1:3,4)-d6*zeroA6(1:3,3);
xCP=pCP0(1); yCP=pCP0(2); zCP=pCP0(3);

% soluzione cinematica inversa braccio articolato 3gdl
% q1
q11=atan2(yCP,xCP)-atan2(0,1);
q12=atan2(yCP,xCP)-atan2(0,-1);
% q3
Bcost=(xCP^2+yCP^2+(zCP-d1)^2-a4^2-d4^2-a3^2)/(2*a3);
q31=atan2(a4,d4)-atan2(Bcost,sqrt(a4^2+d4^2-Bcost^2));
q32=atan2(a4,d4)-atan2(Bcost,-sqrt(a4^2+d4^2-Bcost^2));
% q2
solbr=[q11 NaN q31;
       q11 NaN q32;
       q12 NaN q31;

```

```

    q12 NaN q32];
for i=1:4
    q1=solbr(i,1);
    q3=solbr(i,3);

    s2sys=((cos(q1)*xCP+sin(q1)*yCP)*(cos(q3)*a4-sin(q3)*d4+a3)-(zCP-
d1)*(sin(q3)*a4+cos(q3)*d4))/...
        ((cos(q3)*a4-sin(q3)*d4+a3)^2+(sin(q3)*a4+cos(q3)*d4)^2);
    c2sys=((cos(q1)*xCP+sin(q1)*yCP)*(sin(q3)*a4+cos(q3)*d4)+(zCP-
d1)*(cos(q3)*a4-sin(q3)*d4+a3))/...
        ((cos(q3)*a4-sin(q3)*d4+a3)^2+(sin(q3)*a4+cos(q3)*d4)^2);
    q2=atan2(s2sys,c2sys);
    solbr(i,:)=[q1 q2 q3];
end

% soluzione cinematica inversa polso sferico 3gdl
for i=1:4

    % calcolo matrice polso sr6 wrt sr3
    q1=solbr(i,1);
    q2=solbr(i,2);
    q3=solbr(i,3);
    A3=RobotIRB120.A([1 2 3],[q1 q2 q3]); % sr3 wrt sr0
    A63o=inv(A3)*zeroA6; % matrice polso sr6 wrt sr3

    % soluzione cinematica inversa polso sferico 3gdl
    if A63o(2,3)==1
        q51=0;
        q41=0;
        q61=atan2(-A63o(1,2),A63o(1,1))-q41;
        multi=0;
    elseif A63o(2,3)==-1
        q51=pi;
        q41=0;
        q61=q41-atan2(-A63o(1,2),-A63o(1,1));
        multi=0;
    else
        q51=acos(A63o(2,3));
        q41=atan2(A63o(3,3)/sin(q51),-A63o(1,3)/sin(q51));
        q61=atan2(-A63o(2,2)/sin(q51),A63o(2,1)/sin(q51));
        q52=-acos(A63o(2,3));
        q42=atan2(A63o(3,3)/sin(q52),-A63o(1,3)/sin(q52));
        q62=atan2(-A63o(2,2)/sin(q52),A63o(2,1)/sin(q52));
        multi=1;
    end

    sol6gdl(i,1:6)=[q1 q2 q3 q41 q51 q61];
    if multi==1
        sol6gdl(i+4,1:6)=[q1 q2 q3 q42 q52 q62];
    end
end

end

end

```

**ALLEGATO 5**

```
function [s,s_p,s_pp,tempo]=forma_polinomiale345;

tempo=linspace(0,2,1001);
s= 10*(tempo./2).^3-15*(tempo./2).^4+6*(tempo./2).^5;
s_p= 30*(tempo./2).^2-60*(tempo./2).^3+30*(tempo./2).^4;
s_pp=60*(tempo./2)-180*(tempo./2).^2+120*(tempo./2).^3;

end
```

**ALLEGATO 6**

```

% PIANIFICAZIONE TRAIETTORIA RETTILINEA 3D

clear all
close all
clc

startup_rvc

%% PARAMETRI GEOMETRICI DENAVIT-HARTENBERG
alfa1 = 0 ; a1 = 0 ; d1 = 290e-3 ;
alfa2 = -pi/2 ; a2 = 0 ; d2 = 0 ;
alfa3 = 0 ; a3 = 270e-3 ; d3 = 0 ;
alfa4 = -pi/2 ; a4 = 70e-3 ; d4 = 302e-3 ;
alfa5 = pi/2 ; a5 = 0 ; d5 = 0 ;
alfa6 = -pi/2 ; a6 = 0 ; d6 = 72e-3 ;

alfaT=0; aT=0; dT=0; tetaT0=0;
alfaB=0;aB=0;dB=0;tetaB0=0;

%Parametri tool
incl=60*pi/180;
a=0.06;
b=0.21;

% Forza di reazione dovuta al flusso di fluido uscente dall'ugello
% (intensità costante)
forza=30; % [N]

%% LIMITI DEI GRADI DI LIBERTA' DEI GIUNTI (DA CATALOGO)

qlim1=[-165 165]*pi/180;
qlim2=[-110 110]*pi/180;
qlim3=[-110 70]*pi/180;
qlim4=[-160 160]*pi/180;
qlim5=[-120 120]*pi/180;
qlim6=[-400 400]*pi/180;

%% LINK DEI ROBOT

L(1)=Link('alpha',alfa1,'a',a1,'d',d1,'modified');%,'qlim',qlim1);
L(2)=Link('alpha',alfa2,'a',a2,'d',d2,'offset',-
pi/2,'modified');%,'qlim',qlim2);
L(3)=Link('alpha',alfa3,'a',a3,'d',d3,'modified');%,'qlim',qlim3);
L(4)=Link('alpha',alfa4,'a',a4,'d',d4,'modified');%,'qlim',qlim4);
L(5)=Link('alpha',alfa5,'a',a5,'d',d5,'modified');%,'qlim',qlim5);
L(6)=Link('alpha',alfa6,'a',a6,'d',d6,'modified');%,'qlim',qlim6);

%% COSTRUZIONE DEL ROBOT
ABBirb120=SerialLink(L,'name','ABBirb120');

%% MATRICE eeAtool

eeAtcp=[cos(incl) 0 -sin(incl) -a;
        0 1 0 0;
        sin(incl) 0 cos(incl) b;

```

```

    0 0 0 1];
ABBBirb120.tool=eeAtcp;

%%
qvec=[0 0 0 0 0 0];
[zeroAtcp, A_all]=ABBBirb120.fkine(qvec);
zeroA1=A_all(:, :, 1);
zeroA2=A_all(:, :, 2);
zeroA3=A_all(:, :, 3);
zeroA4=A_all(:, :, 4);
zeroA5=A_all(:, :, 5);
zeroA6=A_all(:, :, 6);
zeroAtcp;
% RobotIRB120.plot(qvec)

%% SPECIFICHE TRAIETTORIA RETTILINEA
deltax=0.08*(-1);
deltay=0.05*(+4);
deltaz=0.03*(+1);
spost_compl=sqrt(deltax^2+deltay^2+deltaz^2);
teta=asin(deltaz/spost_compl);
gamma=asin(deltay/(spost_compl*cos(teta)));

% [s,s_p,s_pp,tempo]=forma_cicloidale;
[s,s_p,s_pp,tempo]=forma_polinomiale345;
figure(1)
plot(tempo,s,tempo,s_p,tempo,s_pp),grid on
xlabel('tempo [s]')
ylabel('Parametro di curva e sue derivate')
legend('s[-]', 's_p[s^-1]', 's_p_p[s^-2]')

zeroAtcp_tempo(:, :, 1)=zeroAtcp;
INVE(:, :, 1)=inv(eeAtcp);
zeroA6(:, :, 1)=zeroAtcp_tempo(:, :, 1)*INVE(:, :, 1);

for i=1:(length(tempo)-1)

    zeroAtcp_tempo(:, :, i+1)=zeroAtcp;
    zeroAtcp_tempo(:, 4, i+1)=zeroAtcp(:, 4)+s(i)*[deltax deltay deltaz 0]';
    INVE(:, :, i+1)=inv(eeAtcp);
    zeroA6(:, :, i+1)=zeroAtcp_tempo(:, :, i+1)*INVE(:, :, i+1);

end

%% CALCOLO DELLE 8 COMBINAZIONI PER PER OGNI ISTANTE DI TEMPO (2s)

for i=1:length(tempo)
    sol6gdl(:, :, i)=cin_inversa(zeroA6(:, :, i));
end

%% CERNITA DEI VETTORI Q

RIGAESATTA=2;

```

```

GRADI_LIBERTA(1,:)=sol6gdl(RIGAESATTA(:,1));
for i=1:(length(tempo)-1)
    for j=1:8
        vettore_norme(j)=norm(sol6gdl(j,:,i+1)-sol6gdl(RIGAESATTA(:,i)));

    end
    [val index]=min(vettore_norme);
    RIGAESATTA=index;
    GRADI_LIBERTA(i+1,:)=sol6gdl(RIGAESATTA(:,i+1));

end
%%

% Vettori gradi di libertà dei giunti nel tempo

q1=GRADI_LIBERTA(:,1);
q2=GRADI_LIBERTA(:,2);
q3=GRADI_LIBERTA(:,3);
q4=GRADI_LIBERTA(:,4);
q5=GRADI_LIBERTA(:,5);
q6=GRADI_LIBERTA(:,6);

%% Calcolo della jacobiana e delle velocità qp

for i=1:length(tempo)

J(:, :, i)=ABBirb120.jacob0([q1(i) q2(i) q3(i) q4(i) q5(i) q6(i)]);
V(:, i)=[s_p(i)*cos(teta)*cos(gamma); s_p(i)*cos(teta)*sin(gamma);
s_p(i)*sin(teta); 0; 0; 0];
qp(:, i)=inv(J(:, :, i))*V(:, i);

end

q1p=qp(1, :)' ;
q2p=qp(2, :)' ;
q3p=qp(3, :)' ;
q4p=qp(4, :)' ;
q5p=qp(5, :)' ;
q6p=qp(6, :)' ;

% PLOTTAGGIO VETTORE DI VELOCITA' GENERALIZZATA

figure(2)

subplot(2,1,1)
plot(tempo,V(1,:),tempo,V(2,:),tempo,V(3,:),tempo,sqrt(V(1,:).^2+V(2,:).^2+V(3,:).^2)),grid on
xlabel('tempo [s]');
title('Velocità Tool Centre Point v_T')
legend('v_T_x [m/s]', 'v_T_y [m/s]', 'v_T_z [m/s]', '|v_T| [m/s]')

subplot(2,1,2)
plot(tempo,V(4,:),tempo,V(5,:),tempo,V(6,:)),grid on
xlabel('tempo [s]');
title('Velocità angolare Tool w_T')

```



```

legend('w_T_x [rad/s]', 'w_T_y [rad/s]', 'w_T_z [rad/s]')

%%

for i=1:length(tempo)
    [zeroAtcp, A_all]=ABBirb120.fkine([q1(i) q2(i) q3(i) q4(i) q5(i) q6(i)]);

    zeropCP(:,i)=A_all(1:3,4,4);
    zeropEE(:,i)=A_all(1:3,4,6);
    zeroTtcp(:,i)=zeroAtcp(1:3,4);
end
%% PLOTTAGGIO TRAIETTORIA E PERCORSO CP

figure (3)

subplot(2,2,1)
plot(tempo,zeropCP(1,:), '-b'),grid on
xlabel('tempo [s]')
ylabel('x_C_P [m]')

subplot(2,2,2)
plot(tempo,zeropCP(2,:), '-b'),grid on
xlabel('tempo [s]')
ylabel('y_C_P [m]')

subplot(2,2,3)
plot(tempo,zeropCP(3,:), '-b'),grid on
xlabel('tempo [s]')
ylabel('z_C_P [m]')

subplot(2,2,4)
plot3(zeropCP(1,:),zeropCP(2,:),zeropCP(3,:), '-k'),grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso CP')

%% PLOTTAGGIO TRAIETTORIA E PERCORSO EE

figure (4)

subplot(2,2,1)
plot(tempo,zeropEE(1,:), '-b'),grid on
xlabel('tempo [s]')
ylabel('x_E_E [m]')

subplot(2,2,2)
plot(tempo,zeropEE(2,:), '-b'),grid on
xlabel('tempo [s]')
ylabel('y_E_E [m]')

subplot(2,2,3)
plot(tempo,zeropEE(3,:), '-b'),grid on

```

```

xlabel('tempo [s]')
ylabel('z_E_E [m]')

subplot(2,2,4)
plot3(zeropEE(1,:),zeropEE(2,:),zeropEE(3,),'-g'),grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso EE')

%% PLOTTAGGIO TRAIETTORIA E PERCORSO TCP
figure (5)

subplot(2,2,1)
plot(tempo,zeropTcp(1,),'-b'),grid on
xlabel('tempo [s]')
ylabel('x_T_C_P [m]')

subplot(2,2,2)
plot(tempo,zeropTcp(2,),'-b'),grid on
xlabel('tempo [s]')
ylabel('y_T_C_P [m]')

subplot(2,2,3)
plot(tempo,zeropTcp(3,),'-b'),grid on
xlabel('tempo [s]')
ylabel('z_T_C_P [m]')

subplot(2,2,4)
plot3(zeropTcp(1,:),zeropTcp(2,:),zeropTcp(3,),'-r'),grid on
xlabel('x')
ylabel('y')
zlabel('z')
title('Percorso TCP')

%% PLOT ROBOT IN MOVIMENTO DURANTE TRAIETTORIA

% figure (6)
figure('units','normalized','outerposition',[0 0 1 1])% Maximizes the figure
window for the figure with handle 5

plot3(zeropCP(1,:),zeropCP(2,:),zeropCP(3,),'-k','Linewidth',2),grid on
view(80,50);
hold on
plot3(zeropEE(1,:),zeropEE(2,:),zeropEE(3,),'-g','Linewidth',2),grid on
plot3(zeropTcp(1,:),zeropTcp(2,:),zeropTcp(3,),'-r','Linewidth',2),grid on
trplot(ABBirb120.base,'frame','A0')
ABBirb120.plot([q1(1:10:end) q2(1:10:end) q3(1:10:end) q4(1:10:end)
q5(1:10:end) q6(1:10:end)], 'notiles','loop','zoom',3)

%% PLOT ROBOT IN CONFIGURAZIONI GENERICHE

figure

```

```
row=500;
qvec_righe=[q1(row) q2(row) q3(row) q4(row) q5(row) q6(row)];
[zeroAtcp, A_all]=RobotIRB120.fkine([q1(row) q2(row) q3(row) q4(row) q5(row)
q6(row)]);

RobotIRB120.plot(qvec_righe,'workspace',[-0.2, 1 -0.3 0.3 0
0.8], 'notiles', 'noname', 'nobase', 'noshadow', 'jointdiam',4.5, 'jointcolor','y', 'linkcolor','c', 'nowrist')
hold on
plot3(zeroCP(1,:),zeroCP(2,:),zeroCP(3,:), '-k', 'Linewidth',2),grid on
plot3(zeroEE(1,:),zeroEE(2,:),zeroEE(3,:), '-g', 'Linewidth',2),grid on
plot3(zeroTcp(1,:),zeroTcp(2,:),zeroTcp(3,:), '-r', 'Linewidth',2),grid on
trplot(RobotIRB120.base, 'length',0.2, 'arrow')
trplot(zeroAtcp, 'length',0.2, 'arrow')
view(0,90)
legend('CP', 'EE', 'TCP')
```