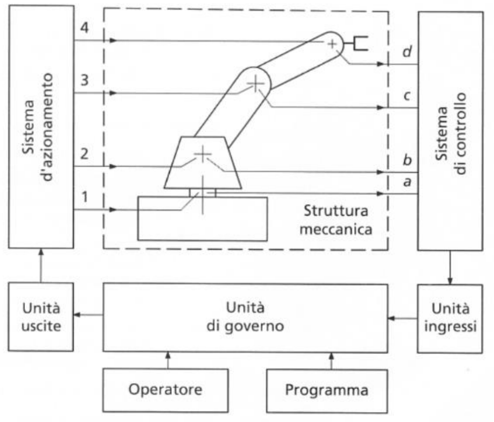
Progetto di robotica industriale

A.A. 2021/2022
Corso di Robotica

Studentessa
Gaia Assunta Bertolino
Matricola 209507

Accenni di robotica industriale

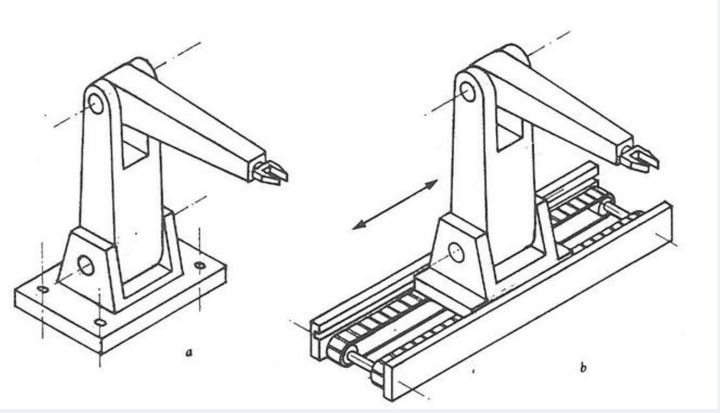
Il sistema dei robot è detto retroazionato ad anello chiuso ovvero è governato da un meccanismo in cui le azioni stesse sono regolate dai feedback delle precedenti azioni rilevati attraverso opportuni sensori presenti sulla struttura stessa del robot.



1 – 2 – 3 – 4: attuatori a - b - c – d: sensori

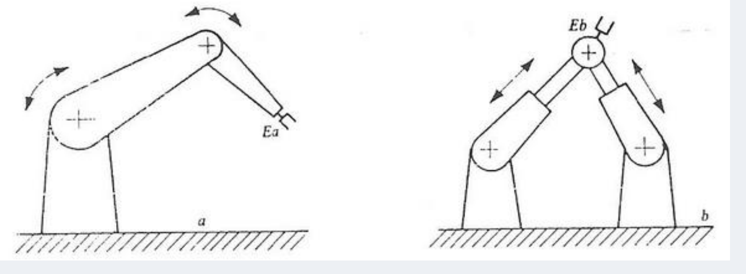
In particolare, col termine industriale si intende quella tipologia di robot solitamente dediti ad azioni automatiche e sostitutive dell'uomo in ambiti, come dice il nome, industriali quali ad esempio operazioni di pick and place. Infatti, tali robot sono costituiti da bracci, detti anche link, e giunti che costituiscono la struttura portante e da una parte finale chiamata end effector che è la diretta responsabile dell'azione automatizzata.
In particolare, la struttura portante è responsabile del posizionamento del robot nello spazio mentre l'end effector (spesso realizzato tramite un polso sferico) realizza l'orientamento del robot. Insieme, posizionamento e orientamento definiscono la cosiddetta posa.

I robot industriali sono solitamente fissi nell'ambiente e cioè ancorato al terreno o allo spazio di lavoro e si distinguono dai robot mobili ovvero quella tipologia che presenta dei sistemi di locomozione.



Tuttavia, la differenza diventa labile nei casi in cui le due caratteristiche vengano unite per realizzare, ad esempio, un robot industriale capace di spostarsi nell'ambiente di lavoro, come ad esempio un robot dell'azienda Amazon capace di operare azioni di pick and place dagli scaffali e di muoversi all'interno del magazzino con autonomia.

I robot industriali si distinguono anche per la struttura seriale o parallela: la prima indica che il collegamento fra i singoli elementi avviene come in una catena mentre la seconda prevede che alcuni link siano collegati a terra e al resto della struttura tramite giunti rotoidali.
La prima tipologia è definita in meccanica catena cinematica aperta e presenta maggiore flessibilità ma meno rigidezza strutturale rispetto alla cosiddetta catena cinematica chiusa corrispondete alla struttura parallela.



Inoltre, ogni spostamento è a sua volta composto da singoli movimenti più semplici, ciascuno dei quali individua un grado di libertà dell'oggetto. Un corpo rigido nello spazio tridimensionale ne ha sei: tre riguardano la posizione, i restanti l'orientamento.

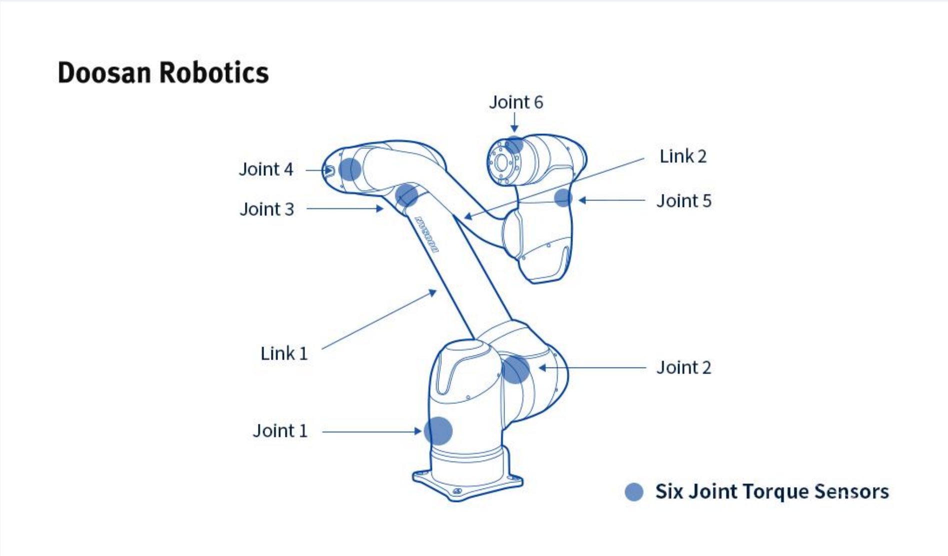
Per quanto riguarda i giunti, esso sono accoppiamenti meccanici fra due link e possono essere di diverse tipologie e con diversi scopi. Due tipi in particolare sono:
- il rotoidale che permette di far ruotare un link
- il prismatico che permette di far traslare un link

Esempio di robot

Un esempio di robot industriale è un cobot della Doosan, azienda sudcoreana.

La parola cobot nasce dalla fusione dei termini inglesi COllaborative e roBOT e sta ad indicare un prodotto della robotica con l'obiettivo di rendere ancora più sinergica la coesistenza fra automi e umani.

Rispetto ai più tradizionali robot industriali, i cobot sono progettati per lavorare insieme alle persone e questo significa che non hanno bisogno di recinti o gabbie di protezione: grazie alle avanzate safety native, all'uso di sensori avanzati e a una programmazione evoluta, infatti, i cobot rispettano distanze di sicurezza tra loro e tra le persone, arrivando a rallentare all'ingresso dell'operatore nell'area di azione e a immobilizzarsi al minimo contatto.



In particolare, il robot presente nell'immagine è dotato di soli due link e ben sei giunti; in questo caso, la massima flessibilità del robot è raggiunta grazie all'utilizzo di giunti di tipologia sferica che permettono un movimento in tutte le direzioni.

Accenni di teoria

Nell'analisi di un problema di robotica, bisogna distinguere fra

- cinematica diretta, ovvero il calcolo della posizione di un oggetto conoscendo le coordinate dei suoi giunti
- cinematica inversa, ovvero la determinazione dei parametri relativi ai giunti per ottenere una posa ricercata

La struttura portante di un robot antropomorfo può essere espressa tramite la convenzione di Denavit-Hartenberg detta anche di D-H che permette di rappresentare una trasformazione nello spazio attraverso un numero mino di parametri che sono quattro ovvero due angoli e due distanze.

Nello spostamento di un robot si può inoltre parlare di velocità introducendo dunque il concetto di cinematica differenziale che riguarda lo studio dei legami fra il moto (cioè la velocità) dei giunti e il modo (cioè la velocità) nello spazio cartesiano. Tale studio viene fatto attraverso il calcolo dello jacobiano, che dal punto di vista matematico è il determinante della matrice di derivate parziali della trasformazione.

Inoltre, le operazioni di traslazione e rotazione si esprimono tramite delle matrici ed è possibile realizzare delle composizioni di esse per esprimere una composizione di movimenti.
Nel caso di due rotazioni successive si moltiplicano le matrici rispettando l'ordine dei movimenti mentre nel caso di una traslazione si orla la matrice di rotazione, che può anche essere pari a quella identità se non vi è alcuna rotazione.

Workspace

Si distinguono due spazi:

- Workspace primario che è lo spazio di lavoro raggiungibile dall'end effector con almeno un orientamento

- Workspace di destrezza che è costituito dai punti raggiungibili dal manipolatore con un orientamento arbitrario dell'end effector

Costruzione in Matlab del robot

```
clear all
global L1 L2 L3 P;

% Dichiarazione della tabella di D-H
% Lunghezza dei bracci
L1 = 0;
L2 = 0.9;
L3 = 0.9;

% Offeset del braccio
d1 = 0;
d2 = 0;
d3 = 0;

% Tipologia di giunto:
% 0 se rotoidale
% 1 se prismatico
sigma1 = 0;
sigma2 = 0;
sigma3 = 0;

% Angolo di twist
alpha1 = pi/2;
alpha2 = 0;
alpha3 = 0;

% Definizione dei link
link1 = Link([0,d1,L1,alpha1,sigma1], 'standard');
link2 = Link([0,d2,L2,alpha2,sigma2], 'standard');
link3 = Link([0,d3,L3,alpha3,sigma3], 'standard');

% Fusione in un'unica struttura dei link
rob = SerialLink([link1 link2 link3], 'name', 'puma 560');
```

In Matlab si ricorre alla libreria Robotics ToolBox realizzata da Peter Corke.
Le funzioni utilizzate nello script sono
- Link la quale riceve sei argomenti relativi ai parametri del braccio rintracciabili nella tabella di D-H
- SerialLink la quale realizza la struttura portante del robot congiungendo i singoli link

In particolare, attraverso il richiamo della funzione SerialLink, la variabile rob conterrà la seguente tabella di *Denavit-Hartenberg*

```
>> rob

rob =

puma 560:: 3 axis, RRR, stdDH, slowRNE
+---+-----------+-----------+-----------+-----------+-----------+
| j |     theta |         d |         a |     alpha |    offset |
+---+-----------+-----------+-----------+-----------+-----------+
|  1|         q1|          0|          0|     1.5708|          0|
|  2|         q2|          0|        0.9|          0|          0|
|  3|         q3|          0|        0.9|          0|          0|
+---+-----------+-----------+-----------+-----------+-----------+
```

Cinematica diretta

Per calcolare la cinematica diretta, si può ricorrere alla moltiplicazione delle matrici omogenee dei giunti. In particolare, è data la matrice di partenza che risulta essere di semplice traslazione.

I seguenti codici fanno uso di variabili simboliche definite tramite il comando syms

Per il giunto R01 la matrice di rototraslazione sarà:

```
R01 = [ cos(theta1)    0     sin(theta1)   0
        sin(theta1)    0    -cos(theta1)   0
              0        1          0        0
              0        0          0        1 ];
```

Per il giunto Rik con k=i-1 ed Li lunghezza del braccio, la matrice di rototraslazione sarà:

```
Rik = [ cos(thetai)    -sen(thetai)    0    Li*cos(thetai)
        sen(thetai)     cos(thetai)    1    Li*sen(thetai)
            0              0           1         0
            0              0           0         1 ];
```

Considerando anche la matrice di traslazione iniziale, la generica funzione di cinematica diretta sarà data dal prodotto Tb3 = Rb0 * R01 * R12 * R23.

In Matlab:

```
syms theta1 theta2 theta3;

L1 = 0;
L2 = 0.9;
L3 = 0.9;

Rb0 =  [1 0 0 0.5;
        0 1 0 0.5;
        0 0 1   1;
        0 0 0   1];

R01 = [ cos(theta1)   0    sin(theta1)  L1*cos(theta1);
        sin(theta1)   0   -cos(theta1)  L1*cos(theta1);
             0        1        0        0;
             0        0        0        1 ];

R12 = [ cos(theta2)     -sin(theta2)    0     L2*cos(theta2);
        sin(theta2)     cos(theta2)     0     L2*sin(theta2);
             0              0           1          0;
             0              0           0          1 ];

R23 = [ cos(theta3)     -sin(theta3)    0     L3*cos(theta3);
         sin(theta3)      cos(theta3)   0     L3*sin(theta3);
             0              0           1          0;
             0              0           0          1 ];



Rn = Rb0*R01*R12*R23;
simplify(Rn)
```

Risultato:

```
[ cos(theta2 + theta3)*cos(theta1), -sin(theta2 + theta3)*cos(theta1),  sin(theta1), (9*cos(theta1)*cos(theta2))/10 - (9*cos(theta1)*sin(theta2)*sin(theta3))/10 + (9*cos(theta1)*cos(theta2)*cos(theta3))/10 + 1/2]
[ cos(theta2 + theta3)*sin(theta1), -sin(theta2 + theta3)*sin(theta1), -cos(theta1), (9*cos(theta2)*sin(theta1))/10 - (9*sin(theta1)*sin(theta2)*sin(theta3))/10 + (9*cos(theta2)*cos(theta3)*sin(theta1))/10 + 1/2]
[             sin(theta2 + theta3),              cos(theta2 + theta3),            0,                                                                  (9*sin(theta2 + theta3))/10 + (9*sin(theta2))/10 + 1]
[                                0,                                 0,            0,                                                                                                                     1]
```

Cinematica inversa e pianificazione della traiettoria

La determinazione delle variabili di giunto conoscendo la posizione terminale dell'end effector è associata alla risoluzione del problema cinematico inverso.

Tuttavia, l’inversione delle equazioni di cinematica diretta non è immediata in quanto esse non sono lineari e, inoltre, la soluzione non è unica. Si parla, ad esempio, di soluzioni a gomito alto o gomito basso.
La soluzione analitica prevede di ricavare gli angoli attraverso le formule del seno e del coseno e a delle sostituzioni successive.
Nella maggior parte dei casi si ricorre invece a metodi numerici come ad esempio il metodo della discesa del gradiente.

Col termine pianificazione della traiettoria si intende invece l’organizzazione delle modalità di evoluzione del moto del manipolatore affinché, da una postura iniziale, giunta ad una finale.

Una volta individuato il cammino da percorrere, si sceglie la legge di moto in base al compito che si è scelto. In particolare, è utilizzata una funzione polinomiale di terzo grado che permette di imporre che l’accelerazione iniziale e quella finale siano pari a 0.

Cinematica differenziale

Nel momento in cui viene introdotto il concetto di coordinate e orientamento che mutano nel tempo si può parlare di velocità.
La velocità si intende come la derivata delle funzioni che esprimono la posa in termini di posizione e orientamento.

La cinematica differenziale viene calcolata attraverso la matrice jacobiana che è la matrice delle derivate parziali della funzione di cinematica diretta.

In particolare, nel caso di un robot antropomorfo la funzione per il calcolo della matrice jacobiana sarà:

```
function J = jacobiano(Q)
global L1 L2 L3
c1 = cos(Q(1));
s1 = sin(Q(1));
c2 = cos(Q(2));
s2 = sin(Q(2));
c3 = cos(Q(3));
s3 = sin(Q(3));

c12 = cos(Q(1) + Q(2));
c23 = cos(Q(2) + Q(3));
s12 = sin(Q(1) + Q(2));
s23 = sin(Q(2) + Q(3));

J = [-s1*(L2*c2 + L3*c23) -c1*(L2*s2 + L3*s23) -L3*c1*s23;
      c1*(L2*c2 + L3*c23) -s1*(L2*s2 + L3*s23) -L3*s1*s23;
      0                    L2*c2 + L3*c23       L3*c23];
end
```

Nel caso del problema, le traiettorie sono due: una triangolare e una circolare.
La traiettoria triangolare è stata suddivida nei tre tratti calcolando per ciascuno la cinematica inversa dei punti e successivamente la cinematica differenziale.

Descrizione del codice Matlab

```matlab
% % Condizione iniziale
Q0 = [0.01; 0.01; 0.01];

% Punti sul cammino da attraversare
P1 = [0.8; 0.8; 0.5];
P2 = [1.2; 0.8; 0.5];
P3 = [1.0; 1.2; 0.5];

% Tempo totale di traiettoria
total = 40;

% TRAIETTORIA TRIANGOLARE

% Primo tratto da P1 a P2

T1 = 0;
T2 = 13.4;
ts = [T1:T2];
sigma = (ts-T1)/(T2-T1);
lambda1 = poly3(sigma);
lambda1d = poly3d(sigma);
N1 = length(lambda1);

% Ciclo di approssimazione
for i = 1:N1
    P = P1 + lambda1(i)*(P2-P1);
    Q = fminsearch('errore_3link',Q0);
    QQ(i,:) = Q;
    PP(i,:) = P;

    J = jacobiano(Q);
    Pd = (P2-P1) * lambda1d(i)/(T2-T1);
    Qd = inv(J)*Pd;
    QQd(i,:) = Qd;
    Q0 = Q;
end

% Secondo tratto da P2 a P3
T3 = 28.4;
ts = [T2:T3];
sigma = (ts-T2)/(T3-T2);
lambda2 = poly3(sigma);
lambda2d = poly3d(sigma);
N2 = length(lambda2);

% Ciclo di approssimazione
for i = 1:N2
    P = P2 + lambda2(i)*(P3-P2);
    Q = fminsearch('errore_3link',Q0);
    QQ(i+N1,:) = Q;
    PP(i+N1,:) = P;

    J = jacobiano(Q);
    Pd = (P3-P2) * lambda2d(i)/(T3-T2);
    Qd = inv(J)*Pd;
    QQd(i+N1,:) = Qd;
    Q0 = Q;
end

% Terzo tratto da P3 a P1
T4 = total;
ts = [T3:T4];
sigma = (ts-T3)/(T4-T3);
lambda3 = poly3(sigma);
lambda3d = poly3d(sigma);
N3 = length(lambda3);
 % Ciclo di approssimazione
 for i = 1:N3
     P = P3 + lambda3(i)*(P1-P3);
     Q = fminsearch('errore_3link',Q0);
     QQ(i+N1+N2,:) = Q;
     PP(i+N1+N2,:) = P;

     J = jacobiano(Q);
     Pd = (P1-P3) * lambda3d(i)/(T1-T3);
     Qd = inv(J)*Pd;
     QQd(i+N1+N2,:) = Qd;
     Q0 = Q;
 end
```

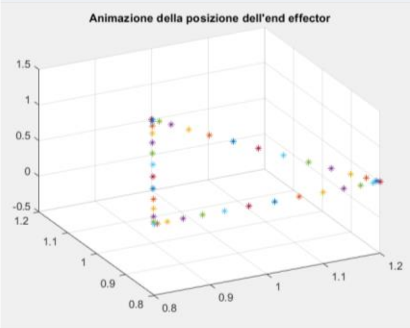
Nel seguente codice sono presenti i tre cicli relativi ai tratti della traiettoria triangolare. Ogni ciclo prevede l'utilizzo del polinomio di terzo grado volto a calcolare il generico punto P che ha la particolarità di essere poco distante dal precedente.
E' in questa parte che si definisce quanto fitta deve essere la discretizzazione.

Inoltre, per ogni punto viene calcolato anche lo Jacobiano che permette di trovare la soluzione al problema della cinematica inversa. Viene così calcolata la velocità in quel punto.

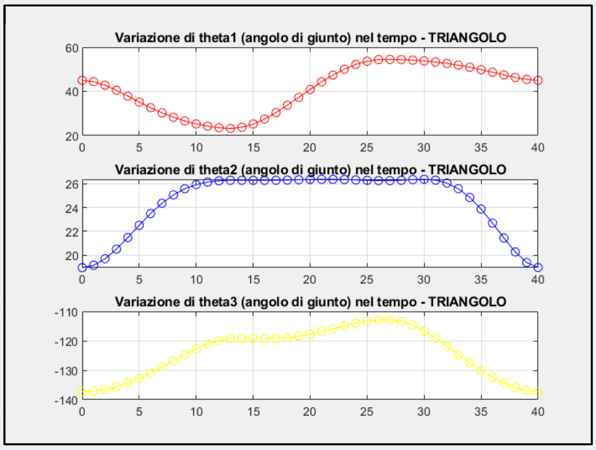
In particolare, i risultati ottenuti, ovvero le coordinate dei punti della traiettoria e della velocità, vengono salvati in una matrice che verrà successivamente utilizzata per la definizione dei grafici.

Tramite il seguente codice si ottiene il grafico accanto (animato su Matlab)

```matlab
% Rappresentazione animata del movimento
figure(2)
plot3(P1(1), P1(2), P1(3), 'o');
hold on
plot3(P2(1), P2(2), P2(3), 'o');
hold on
plot3(P3(1), P3(2), P3(3), 'o');
hold on
for i = 1:N1
plot3(PP(i,1),PP(i,2),PP(i,3),'*')
title('Animazione dell''andamento dell''end effector');
grid on
pause(t/length(PP));
hold on
drawnow
end
hold on
for i = N1+1:N1+N2
plot3(PP(i,1),PP(i,2),PP(i,3),'*')
grid on
pause(t/length(PP));
hold on
drawnow
end
hold on
for i = N1+N2+1:N1+N2+N3
plot3(PP(i,1),PP(i,2),PP(i,3),'*')
grid on
pause(t/length(PP));
hold on
drawnow
end
```

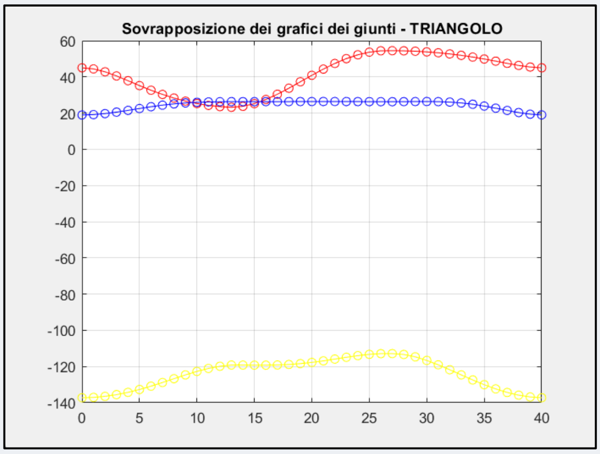


Attraverso il codice sotto si ottengono i grafici riportati a destra che rappresentano i valori delle variabili di giunto che variano nel tempo lungo la traiettoria.

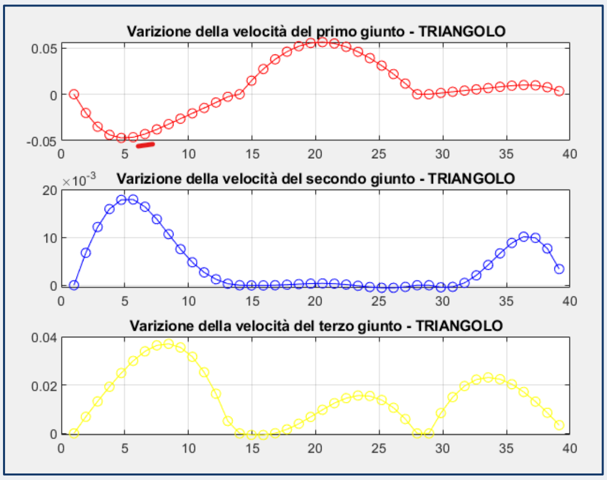


In particolare, sono stati rappresentati sia separatamente che nello stesso plot

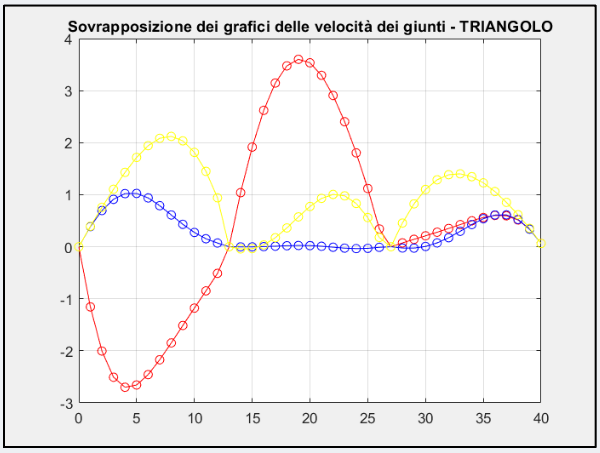
```
passo = total/(length(QQ)+1);
t = [1:passo:total];

% Grafico della variazione dei valori di Q in base a lambda
figure(3)
subplot(3,1,1);
plot(t,QQ(:,1)*180/pi, 'marker','o', 'color','red')
title('Variazione di theta1 (angolo di giunto) nel tempo - TRIANGOLO')
grid

subplot(3,1,2);
plot(t,QQ(:,2)*180/pi,'marker','o', 'color','blue')
title('Variazione di theta2 (angolo di giunto) nel tempo - TRIANGOLO')
grid

subplot(3,1,3);
plot(t,QQ(:,3)*180/pi,'marker','o', 'color','yellow')
title('Variazione di theta3 (angolo di giunto) nel tempo - TRIANGOLO')
grid

figure(4)
plot(t,QQ(:,1)*180/pi, 'marker','o', 'color','red')
hold on
plot(t,QQ(:,2)*180/pi,'marker','o', 'color','blue')
hold on
plot(t,QQ(:,3)*180/pi,'marker','o', 'color','yellow')
grid
title('Sovrapposizione dei grafici dei giunti - TRIANGOLO')
hold off
```



Stessa cosa è stata fatta con le velocità dei giunti.




```
% Grafici velocità
figure(5)
subplot(3,1,1);
plot(t,QQd(:,1), 'marker','o', 'color','red')
title('Varizione della velocità del primo giunto - TRIANGOLO')
grid

subplot(3,1,2);
plot(t,QQd(:,2),'marker','o', 'color','blue')
title('Varizione della velocità del secondo giunto - TRIANGOLO')
grid

subplot(3,1,3);
plot(t,QQd(:,3),'marker','o', 'color','yellow')
title('Varizione della velocità del terzo giunto - TRIANGOLO')
grid

% Sovrapposizione velocità dei giunti
figure(6)
plot(t,QQd(:,1)*180/pi, 'marker','o', 'color','red')
hold on
plot(t,QQd(:,2)*180/pi,'marker','o', 'color','blue')
hold on
plot(t,QQd(:,3)*180/pi,'marker','o', 'color','yellow')
grid
title('Sovrapposizione dei grafici delle velocità dei giunti - TRIANGOLO')
hold off
```



E' interessante notare come la morbidezza delle curve indichi l'andamento senza scatti del robot. Inoltre, nel grafico della velocità si nota come il robot si fermi in concomitanza dei punti della traiettoria in quanto la sua velocità è nulla.

```
% TRAIETTORIA CIRCOLARE

C = [1, 0.95];
R = 0.25;
ts = [0:T4];
sigma = ts/T4;
lambda = poly3(sigma);
lambdad = poly3d(sigma);
N = length(lambda);
diff = (90 + asind(0.2/0.25))*pi/180

for i = 1:N
    theta = lambda(i)*2*pi;
    P = [R*cos(theta-diff) + C(1); R*sin(theta-diff) + C(2); 0.5];
    Q = fminsearch('errore_3link', Q0);
    Pc(i,:) = P;
    Qc(i,:) = Q;

    J = jacobiano(Q);
    Pd = [-R*sin(theta-diff); R*cos(theta-diff); 0];
    Qd = inv(J)*Pd;
    Qcd(i,:) = Qd;
end
```

Per quanto riguarda la traiettoria circolare, il codice sarà similare al precedente con la sostituzione della curva utilizzata che sarà circolare.

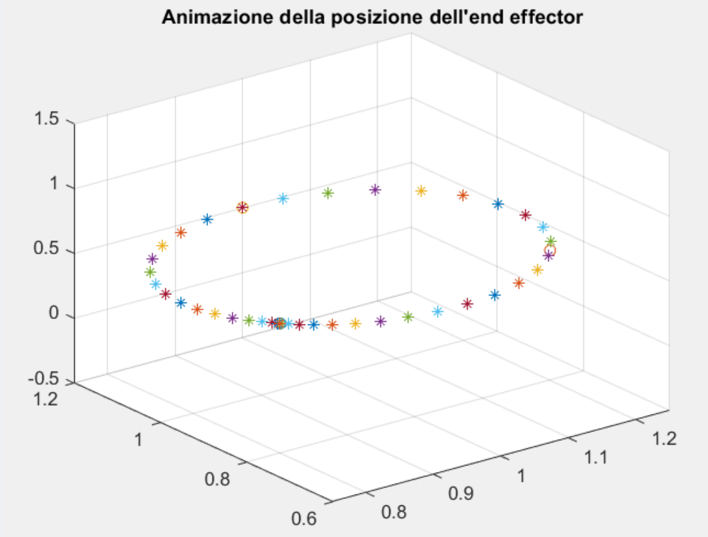
Il calcolo della funzione che descrive la circonferenza si è ottenuto sostituendo le coordinate dei tre punti nella curva generica di equazione $x^2 + y^2 + \alpha x + \beta y + \sigma = 0$

Per far partire il movimento dal punto P1 si è calcolato un valore chiamato *diff* che rappresenta l'angolo della circonferenza che intercetta il punto stesso. Si sottrae dunque tale angolo al valore di theta calcolato in ogni iterazione del ciclo e si sommano le coordinate del centro alle rispettive del punto mentre per la coordinata z, essendo un movimento nullo lungo tale asse, sarà pari al valore costante di 0.5, altezza alla quale si trovano i tre punti.

Rispetto alla traiettoria triangolare, si è ricorso ad un'unica sequenza di movimenti; invece, l'approccio per il calcolo della cinematica inversa e della differenziale è identico.

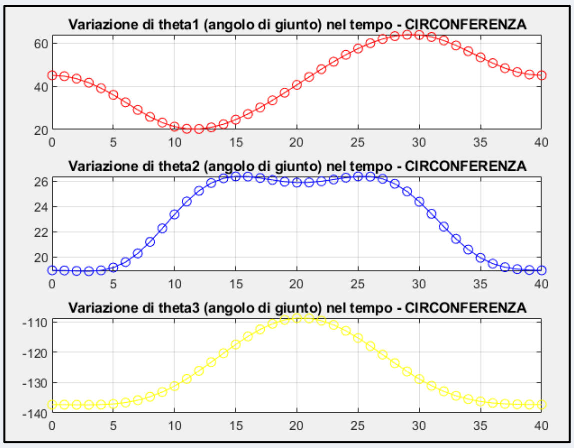
Tramite il seguente codice si ottiene il grafico accanto (animato su Matlab)

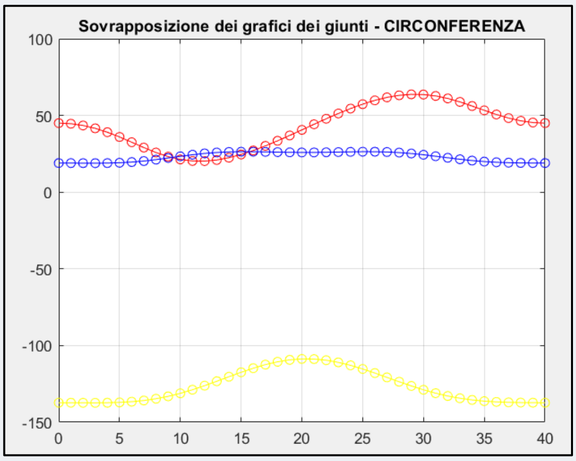
```
% Rappresentazione della traiettoria circolare

figure(7)
plot3(P1(1), P1(2), P1(3), 'o');
hold on
plot3(P2(1), P2(2), P2(3), 'o');
hold on
plot3(P3(1), P3(2), P3(3), 'o');
hold on
for i = 1:length(Pc)
plot3(Pc(i,1),Pc(i,2),0.5,'*')
grid on
pause(t/(length(Pc)));
hold on
drawnow
title('Animazione della posizione dell''end effector')
end
```



Attraverso il codice sotto si ottengono i grafici riportati a destra che rappresentano i valori delle variabili di giunto che variano nel tempo lungo la traiettoria.

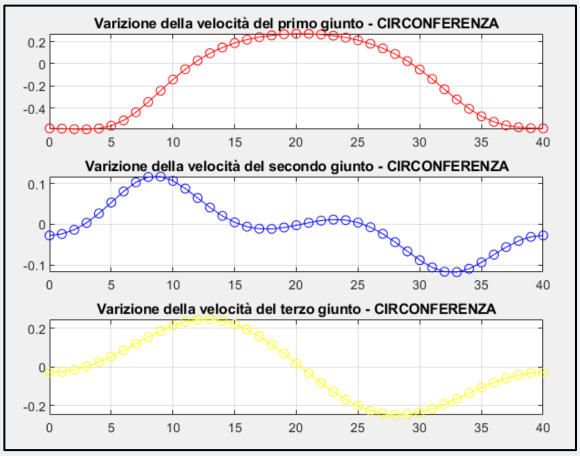
In particolare, sono stati rappresentati sia separatamente che nello stesso plot



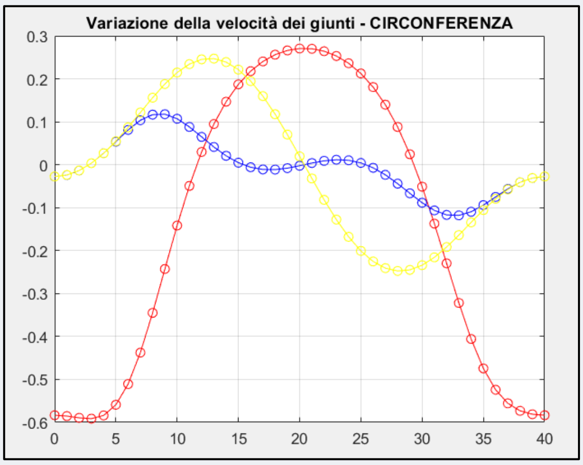



```
passo = total/(length(Qc)+1);
t = [1:passo:total];

% Grafico della variazione dei valori di Q in base a lambda
figure(8)
subplot(3,1,1);
plot(t,Qc(:,1)*180/pi, 'marker','o', 'color','red')
title('Variazione di theta1 (angolo di giunto) nel tempo - CIRCONFERENZA')
grid

subplot(3,1,2);
plot(t,Qc(:,2)*180/pi,'marker','o', 'color','blue')
title('Variazione di theta2 (angolo di giunto) nel tempo - CIRCONFERENZA')
grid

subplot(3,1,3);
plot(t,Qc(:,3)*180/pi,'marker','o', 'color','yellow')
title('Variazione di theta3 (angolo di giunto) nel tempo - CIRCONFERENZA')
grid

figure(9)
plot(t,Qc(:,1)*180/pi, 'marker','o', 'color','red')
hold on
plot(t,Qc(:,2)*180/pi,'marker','o', 'color','blue')
hold on
plot(t,Qc(:,3)*180/pi,'marker','o', 'color','yellow')
grid
title('Sovrapposizione dei grafici dei giunti - CIRCONFERENZA')
hold off
```

Stessa cosa è stata fatta con le velocità dei giunti.



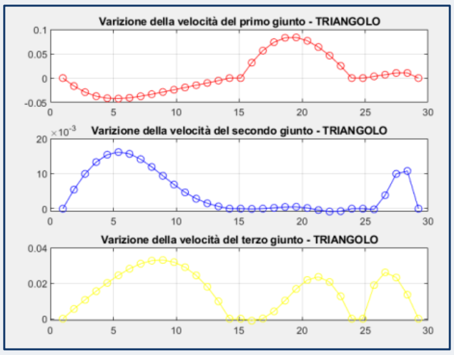

```
% Grafici velocità
figure(10)
subplot(3,1,1);
plot(t,Qcd(:,1), 'marker','o', 'color','red')
title('Varizione della velocità del primo giunto - CIRCONFERENZA')
grid

subplot(3,1,2);
plot(t,Qcd(:,2),'marker','o', 'color','blue')
title('Varizione della velocità del secondo giunto - CIRCONFERENZA')
grid

subplot(3,1,3);
plot(t,Qcd(:,3),'marker','o', 'color','yellow')
title('Varizione della velocità del terzo giunto - CIRCONFERENZA')
grid

% Sovrapposizione grafici velocità
figure(11)
plot(t,Qcd(:,1), 'marker','o', 'color','red')
title('Variazione della velocità dei giunti - CIRCONFERENZA')
grid
hold on
plot(t,Qcd(:,2), 'marker','o', 'color','blue')
hold on
plot(t,Qcd(:,3), 'marker','o', 'color','yellow')
hold off
```

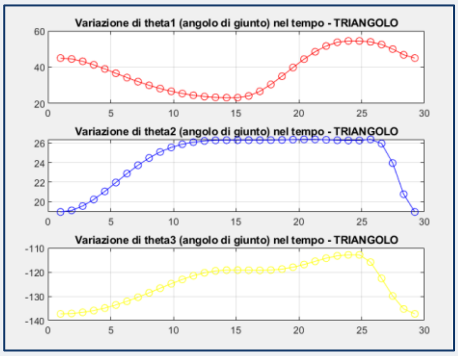


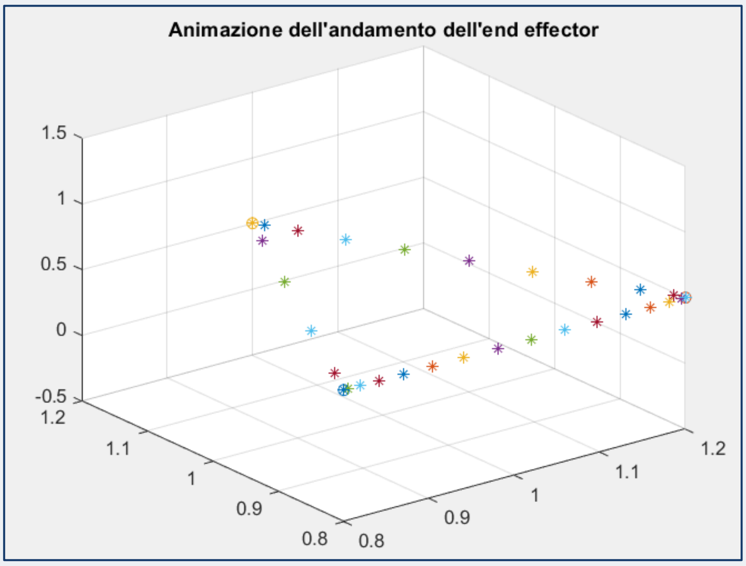
E' interessante notare come, anche in questo caso, la morbidezza delle curve indichi l'andamento senza scatti del robot.
Inoltre, nel grafico della velocità si nota come il robot si fermi in concomitanza dei punti della traiettoria in quanto la sua velocità è nulla.

Come primo esempio all'interno dello script sono stati impostati un tempo totale di percorrenza pari a 40 s e nella traiettoria triangolare, tre segmenti con rispettivi tempi di inizio percorrenza pari a 13.4 s e 28.4 s oltre allo start posto a 0 s.
Modificando questi tre dati nello script (tempo totale e tempi intermedi per la traiettoria triangolare) è possibile notare come i tempi siano diversi e di conseguenza anche le velocità.
Ad esempio, con tempo di percorrenza pari a 30 s e tempi intermedi pari a 15 s e 25 s si avranno i seguenti grafici:

TRAIETTORIA TRIANGOLARE







TRAIETTORIA CIRCOLARE