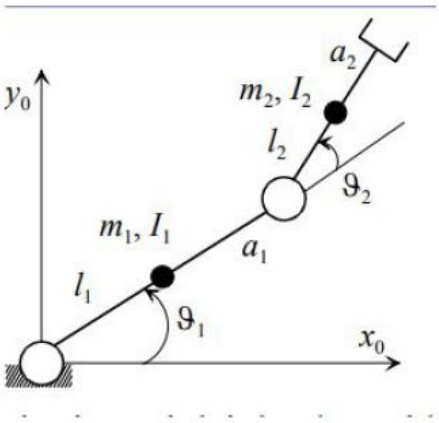
Relazione Progetto Robotica aa: 2022/2023

A cura di Giuseppe Coppola mat: 219811

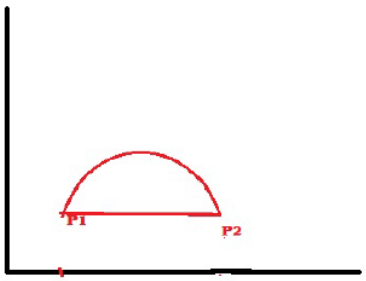
Dati:



Prendiamo in considerazione un robot planare avente i seguenti parametri:

link	a_i (m)	m_i (Kg)	l_i (m)	I_i $(kg\,m^2)$
1	1.0	1.2	0.5	$\frac{1}{10}$
2	0.8	0.8	0.4	$\frac{16}{325}$

Obiettivo: il robot deve seguire il seguente percorso:



Con $P_1 = [0.5\ 0.3]^T$ e $P_2 = [1.5\ 0.3]^T$; il tratto P_1 -> P_2 segue un movimento rettilineo mentre P_2 -> P_1 segue una semicirconferenza.

Ogni tratto viene effettuato in degli istanti temporali: il tratto P_1 -> P_2 in 10 secondi, mentre P_2 -> P_1 in 20 secondi.

Lo script matlab che segue va a definire i vari parametri utilizzati all'interno del sistema:

Parametri.m

```
clear
%Lunghezza dei bracci
a1 = 1;
a2 = 0.8;
L = [a1;a2];

%Masse
m1 = 1.2;
m2 = 0.8;

%Asse di rotazione
l1 = 0.5;
l2 = 0.4;

%Inerzie
I1 = 1/10;
I2 = 16/325;

%Istanti di tempo
T = [10;30];

%Punti
P1 = [0.5; 0.3];
P2 = [1.5; 0.3];

%Gravità
g = 9.8;

%Condizioni iniziali
Q = twolink_inverse(L,P1);
q10 = Q(1);
q20 = Q(2);

%Calcolo dei parametri dinamici
bc11 = I1+I2+m1*l1^2+m2*(a1^2+l2^2);
bc12 = I2+m2*l2^2;
bc21 = bc12;
bc22 = m2*l2^2+I2;
BC = [bc11 bc12; bc21 bc22];

bv11 = 2*a1*m2*l2;
bv12 = a1*m1*l2;
bv21 = bv12;
bv22 = 0;
BV = [bv11 bv12; bv21 bv22];

H = -m2*a1*l2;

G1 = (m1*l1+m2*a1)*g;
G2 = m2*l2*g;
```

Oltre ai parametri descritti all'interno della tabella, viene istanziata la forza di gravità **g**.

Per quanto riguarda le **condizioni iniziali**, dei giunti q_1 e q_2, nello spazio dei giunti, vengono calcolate tramite la funzione *twolink_inverse* che effettua il calcolo della cinematica inversa (passando come parametri in ingresso alla funzione le masse dei bracci e il punto P_1).

Per il calcolo dei **parametri dinamici**, si deve soddisfare la seguente equazione dinamica del robot:

$$T(t) = B(Q)\ddot{Q} + h(Q,\dot{Q})\dot{Q} + G(Q)$$

ovvero un'equazione differenziale del secondo ordine.

Quindi:

- $\boldsymbol{B(Q)}$ in questo caso dipende solamente da q_2:

$$B(q_2) = BC + \cos(q_2)\, BV$$

dove *BC* e *BV* vengono calcolate tramite l'approccio Lagrangiano

- $\boldsymbol{h(Q,\dot{Q})}$ si ricava effettuando delle derivate alla matrice B tramite i *coefficienti di Cristoffel*.

$$h(Q,\dot{Q}) = \begin{pmatrix} H\sin(q_2)\,[2\dot{q_1}\dot{q_2} + \dot{q_2^2}\,] \\ H\sin(q_2)\dot{q_1^2} \end{pmatrix}$$

dove $H = -m_2 a_1 l_2$

La prima cosa da gestire è il **generatore di riferimenti**.

Per ottenere il generatore di riferimenti bisogna come prima cosa cercare il legame $\boldsymbol{\lambda = \lambda(\sigma)}$, così da andare a mappare la funzione considerata.

Come prima cosa si definisce un generico tempo **t**; definiti quindi $\mathbf{t_1}$ e $\mathbf{t_2}$:

$$t = t_1 + \sigma(t_2 - t_1)$$

e quindi:

$$\sigma = \frac{t - t_1}{t_2 - t_1}$$

Se si considera una generica funzione **f**, la sua mappatura risulta essere:

$$f = f_1 + \lambda(\sigma)[f_2 - f_1]$$

ovvero nel dominio del tempo

$$f(t) = f_1 + \lambda(\frac{t - t_1}{t_2 - t_1})(f_2 - f_1)$$

Se si colcola la derivata di *f(t)*:

$$f'(t) = \frac{f_2 - f_1}{t_2 - t_1} \lambda'(\frac{t - t_1}{t_2 - t_1})$$

dove $\frac{f_2 - f_1}{t_2 - t_1}$ rappresenta la velocità media della funzione **f**, mentre λ' viene chiamato **fattore scala** ed è un fattore che tende a modificare la velocità.

Il quesito più importante è come si deve progettare $\boldsymbol{\lambda(\sigma)}$.

In generale si usano polinomi di terzo grado, quinto grado o settimo grado. In questo robot viene usato l'approccio con i *polinomi di settimo grado*:

$$\lambda(\sigma) = a\,\sigma^7 + b\,\sigma^6 + c\,\sigma^5 + d\,\sigma^4 + e\,\sigma^3 + f\,\sigma^2 + g\,\sigma + h$$

con i seguenti vincoli, garantendo continuità su funzione, derivata prima, derivata seconda e derivata terza:

$$\lambda(0) = 0 \qquad \lambda(1) = 1$$
$$\lambda'(0) = 0 \qquad \lambda'(1) = 0$$
$$\lambda''(0) = 0 \qquad \lambda''(1) = 0$$
$$\lambda'''(0) = 0 \qquad \lambda'''(1) = 0$$

Bisogna quindi risolvere il seguente sistema di equazioni lineari:

$$\begin{cases} h=0 \\ a+b+c+d+e+f+g+h=1 \\ g=0 \\ 7a+6b+5c+4d=0 \\ f=0 \\ 42a+30b+20c+12d=0 \\ e=0 \\ 210a+120b+60c+24d=0 \end{cases}$$

Trovati i vari coefficienti si arriva ad un *polinomio di settimo grado* del tipo:

$$\lambda(\sigma) = -20\sigma^7 + 70\sigma^6 - 84\sigma^5 + 35\sigma^4$$

A questo punto, bisogna passare alla progettazione del **generatore di** riferimenti; l'algoritmo per costruirlo funziona in questo modo:

- al generico istante di tempo **t**, noto $\boldsymbol{\sigma}$:
  $$\sigma = \frac{t - t_1}{t_2 - t_1}$$
  si può calcolare $\boldsymbol{\lambda(t)}$ tramite il polinomio in questione.

- Si ha che $\boldsymbol{q}$ e $\boldsymbol{\dot{q}}$ di riferimento valgono:

$$\begin{cases} q^*(t) = q_1 + \lambda(\sigma)(q_2 - q_1) \\ \dot{q^*}(t) = \lambda'(\sigma) \dfrac{q_2 - q_1}{t_2 - t_1} \end{cases}$$

Seguendo l'algoritmo di costruzione del generatore di riferimenti, dopo aver progettato $\boldsymbol{\lambda(t)}$ si passa alla parametrizzazione della funzione in questione; in questo caso si deve effettuare una parametrizzazione di una retta e di una circonferenza.

Per quanto riguarda la retta e quindi il tratto *rettilineo*:

- Senza la variabile del tempo:

$$\begin{cases} P(\lambda) = P_1 + \lambda(P_2 - P_1) \\ 0 \leq \lambda \leq 1 \end{cases}$$

- Con la variabile del tempo:

$$P(\lambda(t)) = P_1 + \lambda(t)(P_2 - P_1)$$

  dove $\boldsymbol{\lambda(t)}$ è il polinomio di settimo grado progettato in precedenza.

Nel caso di questo movimento effettuato dal robot, l'unico componente del tratto rettilineo a variare è quella delle ascisse.

Invece, per quanto riguarda la circonferenza e quindi il tratto della *semicirconferenza*:

- La parametrizzazione avviene in due fasi:
  - Fase 1: si progetta la variazione dell'angolo:

$$\theta\big(\lambda(t)\big) = \theta_0 + \lambda(t)(\theta_1 - \theta_0)$$

    dove $\boldsymbol{\lambda(t)}$ è sempre il polinomio di settimo grado già progettato.

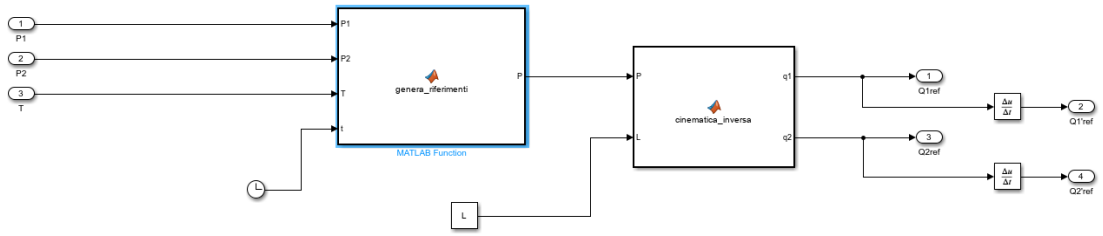
  - Fase 2: dato il centro della circonferenza $\boldsymbol{c} = (\boldsymbol{c_x}, \boldsymbol{c_y})$ e il raggio $\boldsymbol{r}$:

$$P_x = c_x + r * \cos(\theta(\lambda(t))$$
$$P_y = c_y + r * \sin(\theta(\lambda(t))$$

Segue qui lo script matlab che realizza il **generatore di riferimenti**:

```
function P = genera_riferimenti(P1,P2,T,t)

t1 = T(1);
t2 = T(2);
P1x = P1(1);
P1y = P1(2);
P2x = P2(1);
P2y = P2(2);

P=[0;0];

cx = 1;
cy = 0.3;
r = 0.5;


if ( t>t2 )
    P(1) = P1x;
    P(2) = P1y;

end

if ( t1>=t )
    sigma_r = (t-0)/(t1-0);
    lambda_r = -20*sigma_r^7+70*sigma_r^6-84*sigma_r^5+35*sigma_r^4;
    P(1) = P1x + lambda_r*(P2x - P1x);
    P(2) = P1y + lambda_r*(P2y - P1y);

end

if ( t>t1 && t<=t2)
    sigma_c = (t-t1)/(t2-t1);
    lambda_c = -20*sigma_c^7+70*sigma_c^6-84*sigma_c^5+35*sigma_c^4;
    theta0 = 0;
    theta1 = pi;
    theta = theta0 + lambda_c*(theta1-theta0);
    P(1) = cx + r*cos(theta);
    P(2) = cy + r*sin(theta);
end

end
```

Il generatore di riferimenti, in *simulink*, viene messo all'interno del seguente sottoinsieme:



Si distinguono due blocchi:

- **Genera riferimenti**: tutto ciò che si è descritto in precedenza
- **Cinematica inversa**: trasforma i riferimenti dallo spazio di lavoro allo *spazio dei giunti*, ovvero:

  $$Q(\lambda(t))$$

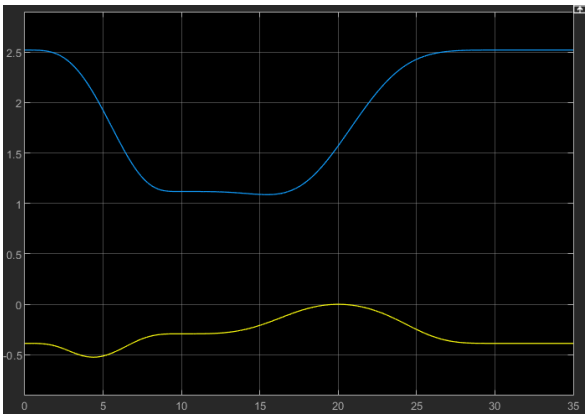
  Se si effettua la derivata:

  $$\dot{Q} = \frac{\vartheta Q}{\vartheta \lambda}\,\dot{\lambda}$$

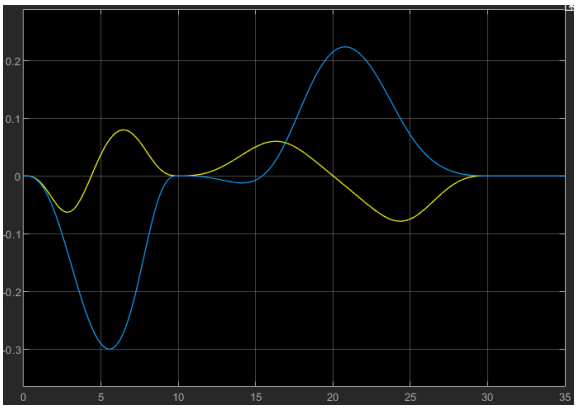
  questo sta a significare che tutto dipende dal modo in cui si sceglie $\boldsymbol{\lambda}$.

Si nota che P1, P2 e T sono le costanti in ingresso al blocco, ma quest'ultime corrispondono alle costanti di ingresso al robot in questione.

I riferimenti che vengono generati sono i seguenti:



$\boldsymbol{q^*(t)}$

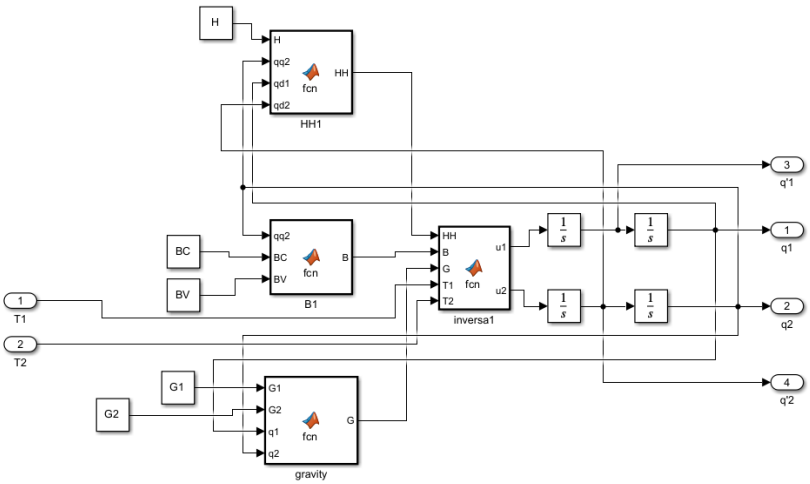


$\boldsymbol{\dot{q}^*(t)}$

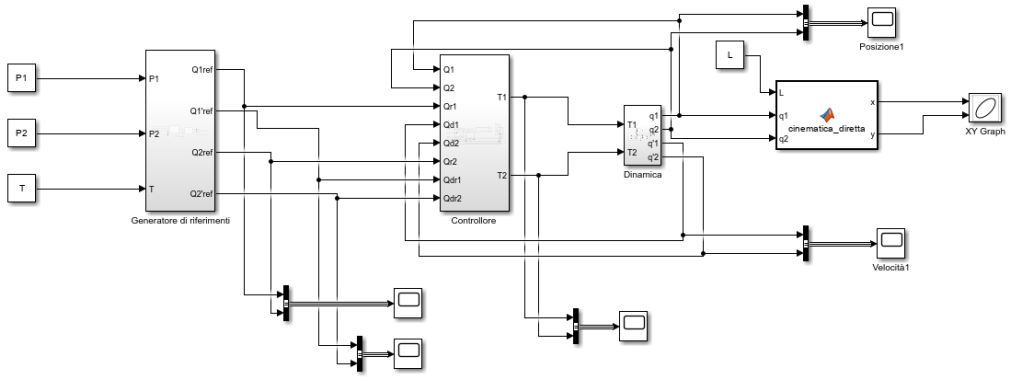
Il motivo per cui si usa un polinomio di settimo grado è quello che si hanno movimenti più "docili" da parte del robot.

Per quanto riguarda, invece, l'**equazione dinamica del robot**, in *simulink***,** viene descritta nel seguente sottosistema:

$$T(t) = B(Q)\ddot{Q} + h(Q,\dot{Q})\dot{Q} + G(Q)$$



Quest'ultimo è uguale in tutte le tipologie di robot descritte tranne in quello in cui non vi è presenza di gravità.



Composizione del robot in simulink

Le due componenti, **generatore di riferimenti** e blocco che descrive l'**equazione dinamica del robot**, rimangono inviariati per le varie tipologie di robot; l'unico componente che varia è il **controllore**. Distinguiamo varie tipologie di robot:

Robot con controllore PD in assenza di gravità

In questo caso, all'interno del sottoinsieme che descrive la dinamica del robot non è presente il calcolo del vettore gravità.

Il *controllore PD*, come gli altri tipi di controllori descritti in questo robot, è un classico **controllo in retroazione** e quindi un controllo a *ciclo chiuso* poiché vengono inseriti sensori capaci di misurare istante per istante l'andamento delle variabili di giunto.

In generale il controllore PD descrive:

$$f(t) = K_P[x^*(t) - x(t)] + K_D[\dot{x^*}(t) - \dot{x}(t)]$$

dove K_P (guadagno proporzionale) e K_D (guadagno derivativo) sono sempre positivi e portano alla **stabilità del sistema**.

Quindi, sorge il problema con le rotazioni poiché, per effetto di quest'ultime, entrano in gioco le coppie di forze **T** agenti su ogni giunto.

Allora, se il generatore di riferimenti genera $\boldsymbol{q^*(t)}$ e $\boldsymbol{\dot{q^*}(t)}$ il controllore agisce in questo modo:

$$T_i(t) = K_{P_i}\left[q^*_i(t) - q_i(t)\right] + K_{D_i}\left[\dot{q^*_i}(t) - \dot{q_i}(t)\right]$$

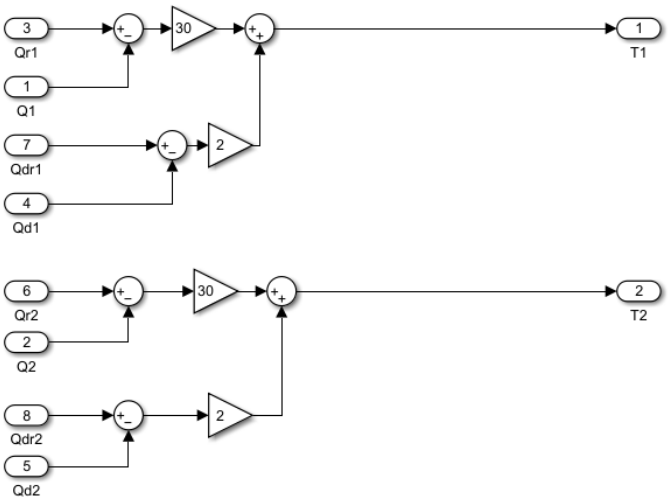
dove:

$$c_{P_i} = q^*_i(t) - q_i(t)$$

$$c_{V_i} = \dot{q^*_i}(t) - \dot{q_i}(t)$$

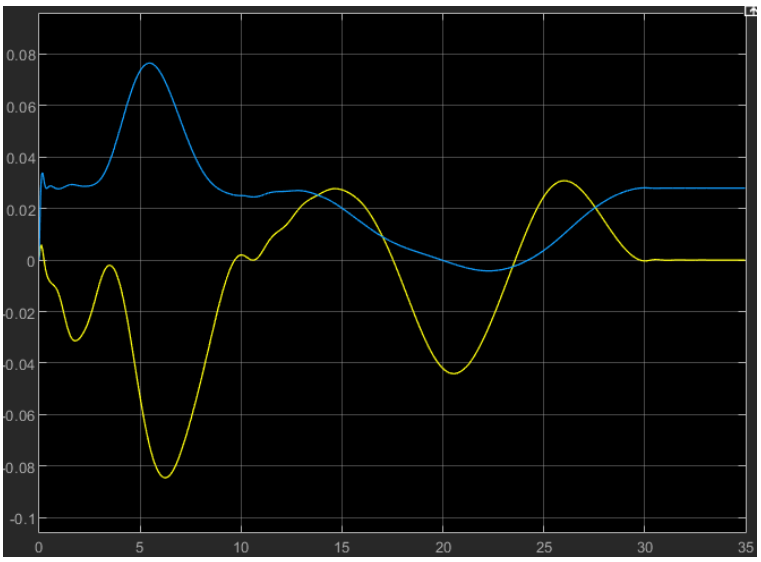
L'obiettivo del controllore è quello di portare a zero $\boldsymbol{c_{P_i}}$ e $\boldsymbol{c_{V_i}}$.

In *simulink* è descritto dal blocco seguente:



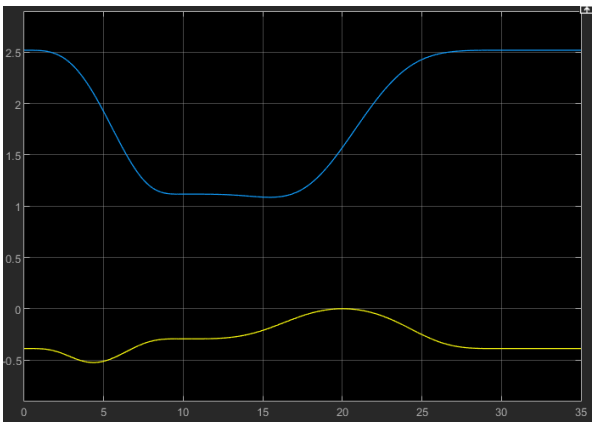
Scegliendo appositamente $\boldsymbol{K_P = 30}$ e $\boldsymbol{K_D = 2}$ così che le forze generate diano il giusto movimento al robot.

Le coppie T, in questo caso $\boldsymbol{T_1}$ e $\boldsymbol{T_2}$, risultano:

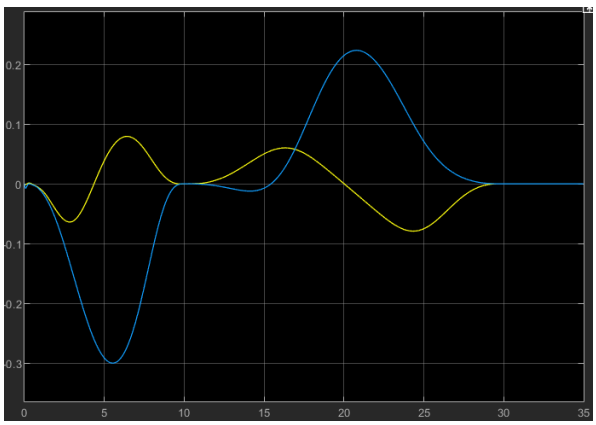


Si può notare che sono coppie abbastanza piccole poiché non essendo presente la forza di gravità devono solamente contrastare le **forze apparenti**; infatti oscillano da un valore massimo di $\boldsymbol{0.08\ N \cdot m}$ ad un valore minimo di $\boldsymbol{-0.08\ N \cdot m}$.

Una volta generate le forze, il blocco che descrive la dinamica del robot calcola il movimento effettivo; nello spazio dei giunti:

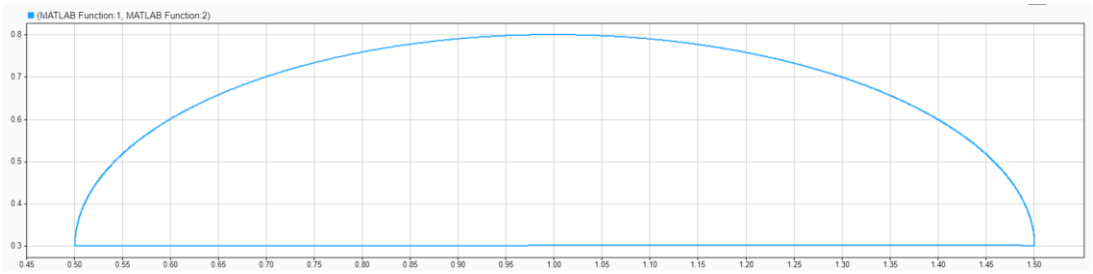


$\boldsymbol{q(t)}$



$\boldsymbol{q(\dot{t})}$

Invece, nello spazio di lavoro:



Si notano errori abbastanza piccoli durante il movimeto rettilineo.

Robot con controllore PID in presenza di gravità

Il controllore, in questo caso, descrive:

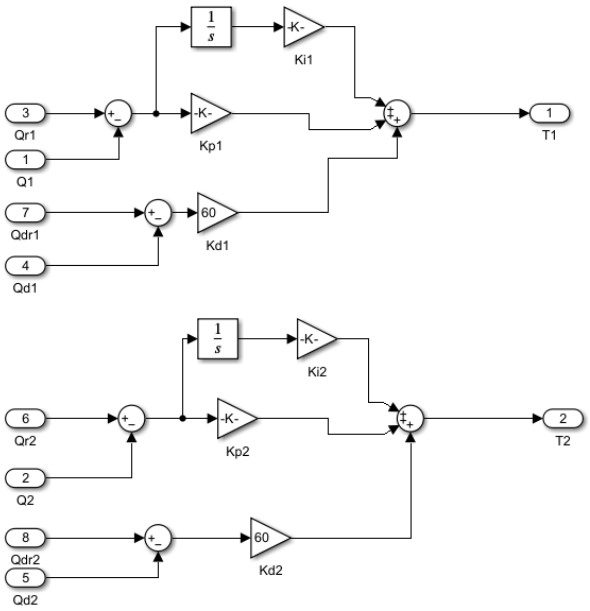
$$T_i(t) = \{K_{P_i}[q^*{}_i(t) - q_i(t)] + K_{D_i}[\dot{q^*{}_i}(t) - q_i(\dot{t})]\} + K_{i_i}\int_0^t [\, q^*{}_i(\sigma) - q_i(\sigma)]\, d\sigma$$

dove $\boldsymbol{K_i}$ è definita positiva.

Si effettua questa scelta per portare l'errore a zero in presenza di gravità; infatti, per compensare la gravità:

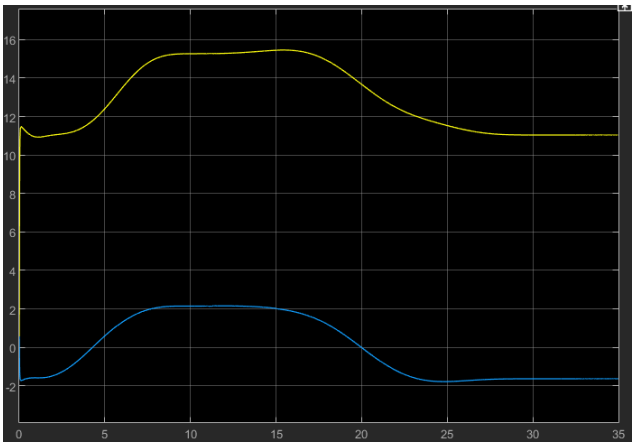
$$\lim_{t\to\infty} q(t) = q^*$$

In *simulink*, il controllore PID viene descritto dal seguente blocco:



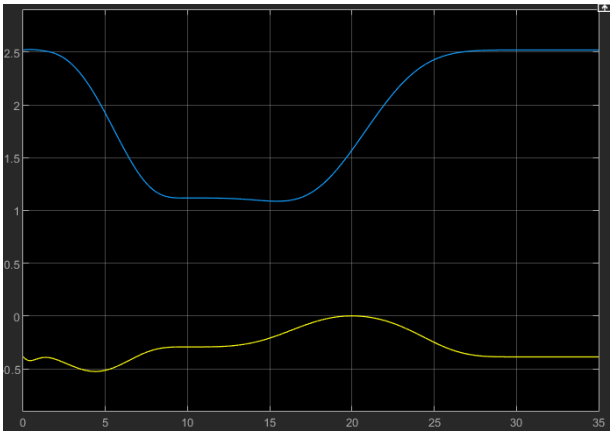
Scegliendo appositamente $\boldsymbol{K_P = 175}$, $\boldsymbol{K_D = 60}$ e $\boldsymbol{K_I = 500}$

Quest'ultimo genera la seguente coppia:

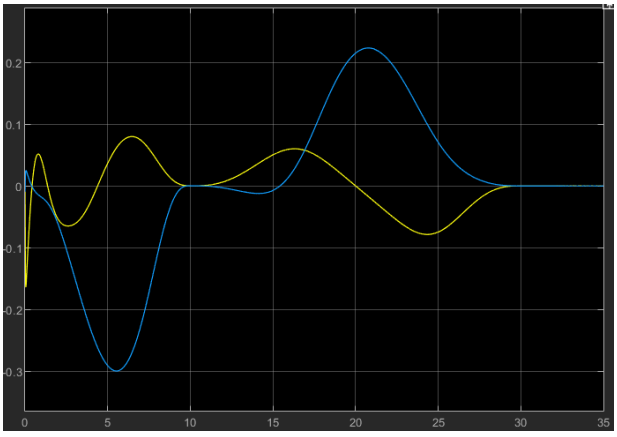


Si nota l'improvvisa variazione nell'istante iniziale proprio perché il controllore cerca di compensare la gravità.

Le variabili di giunto calcolate nello spazio dei giunti:

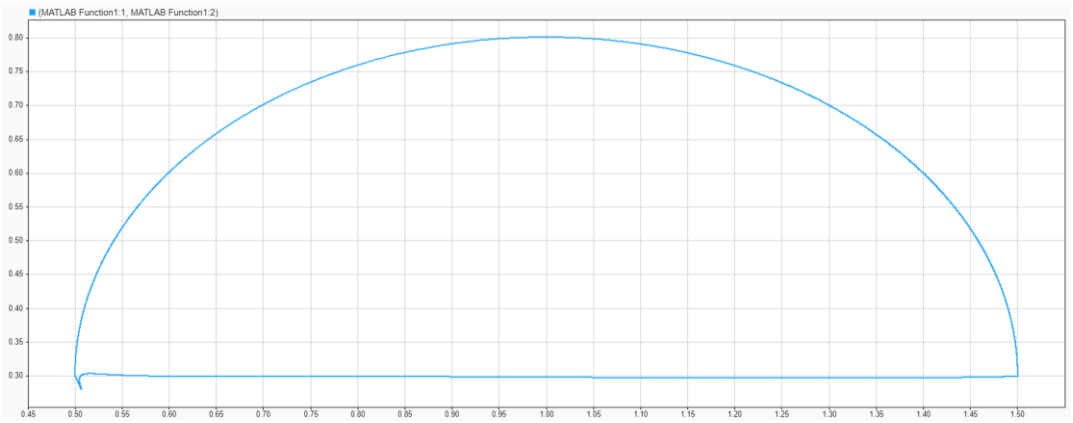


$\boldsymbol{q(t)}$



$\boldsymbol{\dot{q}(t)}$

Nello spazio di lavoro il percorso risulta essere:



Si può notare un lieve errore nel punto iniziale del tratto rettilineo.

Robot con controllore non lineare in presenza di gravità

Questi tipi di controllori non lineari sono quei controllori in cui compare un termino non lineare nella retroazione.

In questo caso ne vengono gestiti di diversi tipi dove i termini non lineari saranno il vettore delle forze apparenti $\boldsymbol{h}(\boldsymbol{Q}, \dot{\boldsymbol{Q}})\dot{\boldsymbol{Q}}$ e il vettore delle forze di gravità $\boldsymbol{G}(\boldsymbol{Q})$.

Vengono distinti anche i casi in cui la compensazione avviene perfettamente, ovvero il controllore conosce perfettamente le masse dei bracci, e non perfettamente, ovvero il controllore conosce le masse dei bracci ma non sono misurate in maniera perfetta.

Controllore con compensazione della gravità

Compensazione perfetta

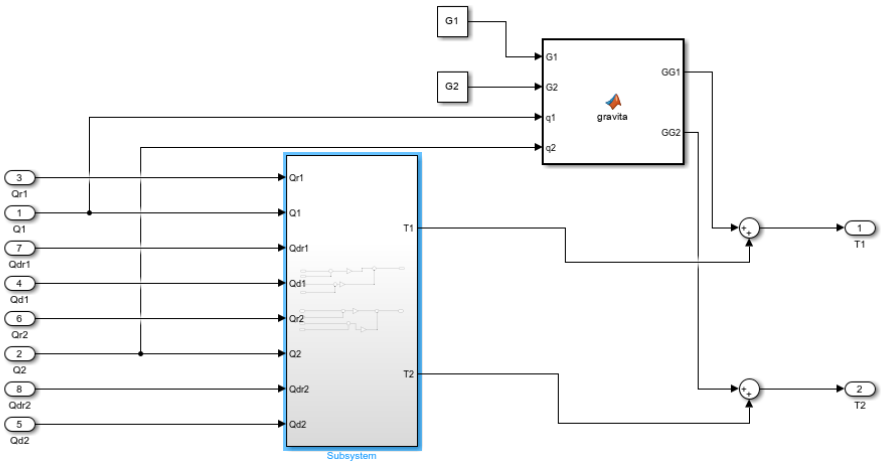
Il controllore in questione descrive:

$$T_i(t) = K_{P_i}\left[q^*{}_i(t) - q_i(t)\right] + K_{D_i}\left[q^*{}_i\dot{}(t) - q_i\dot{}(t)\right] + G(Q)$$

Questo va a compensare perfettamente il vettore gravità perché:

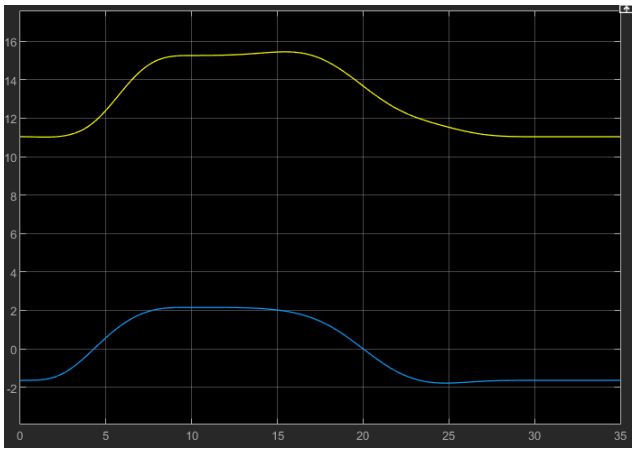
$$B(Q)\ddot{Q} + h(Q,\dot{Q})\dot{Q} + \cancel{G(Q)} = K_{P_i}\left[q^*{}_i(t) - q_i(t)\right] + K_{D_i}\left[q^*{}_i\dot{}(t) - q_i\dot{}(t)\right] + \cancel{G(Q)}$$

In *simulink*:



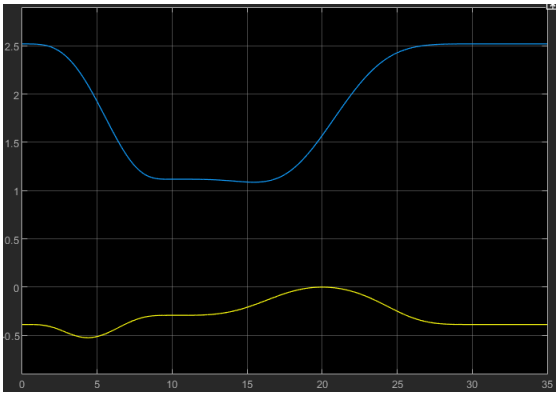
In questo caso, $\boldsymbol{K_P = 200}$ e $\boldsymbol{K_D = 21}$.

Se si osserva l'andamento delle coppie:

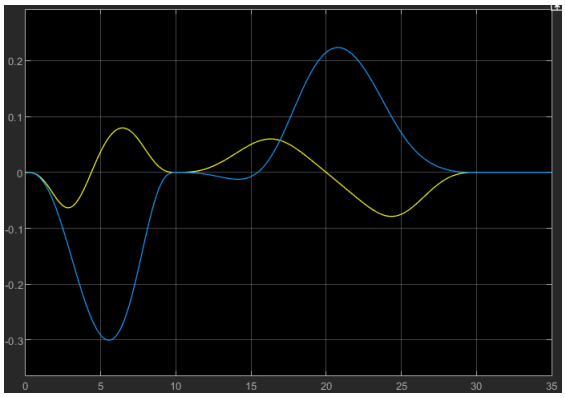


Si nota che per $\boldsymbol{t} \rightarrow \infty$ le coppie tendono al loro valore iniziale proprio perchè il robot stesso torna nella sua posizione di partenza.

Se si osserva il movimento effettivo del robot nello spazio dei giunti, dopo aver passato la coppia al blocco che calcola la dinamica del robot, risulta essere:

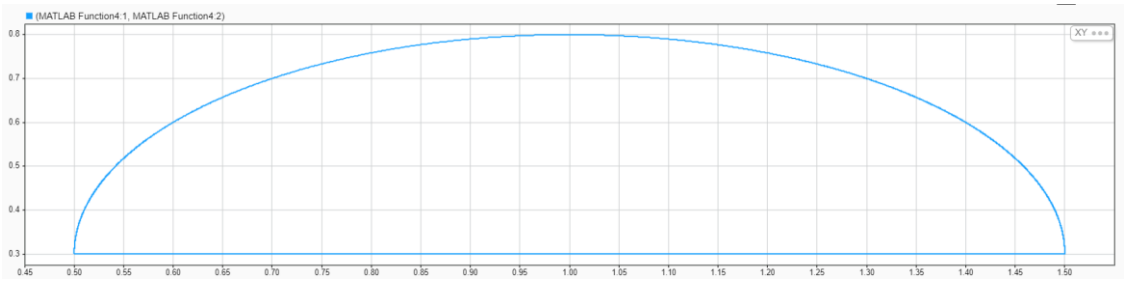


$\boldsymbol{q(t)}$



$\boldsymbol{q(\dot{t})}$

Nello spazio di lavoro:



Essendo una compensazione perfetta non si notano nemmeno errori nello spostamento effettivo del robot.

Compensazione non perfetta

Essendo la compensazione non perfetta, il controllore conosce delle *masse fittizie*; in questo caso queste masse risultano essere:

$$m_1 = 1.15$$

$$m_2 = 0.73$$

La variazione rispetto alle masse effettive è di circa 0.07.

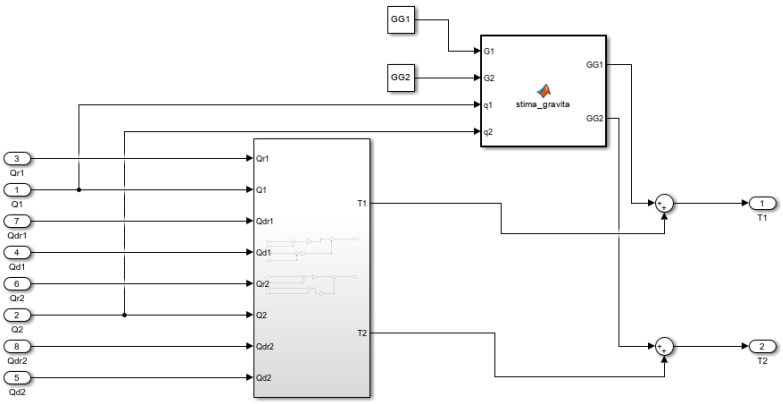
Quindi, il controllore descrive:

$$T_i(t) = K_{P_i}\left[{q^*}_i(t) - q_i(t)\right] + K_{D_i}\left[\dot{{q^*}_i}(t) - \dot{q_i}(t)\right] + \hat{G}(Q)$$

dove $\widehat{\boldsymbol{G}}(\boldsymbol{Q})$ è la stima del vettore gravità.

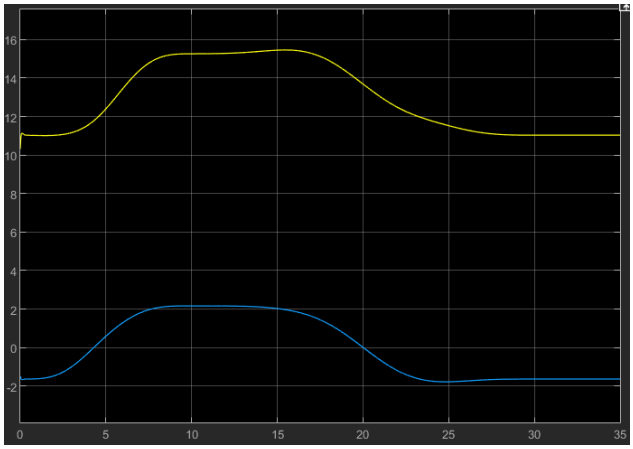
In questo caso non si compensa completamente la gravità.

In *simulink*:



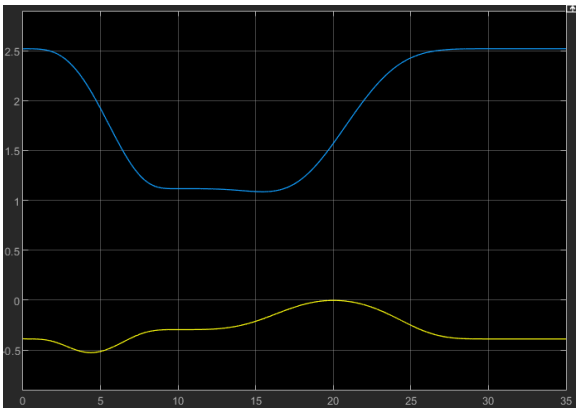
Si scelgono $\boldsymbol{K_P = 325}$ e $\boldsymbol{K_D = 31}$.

Se si osservano le coppie:

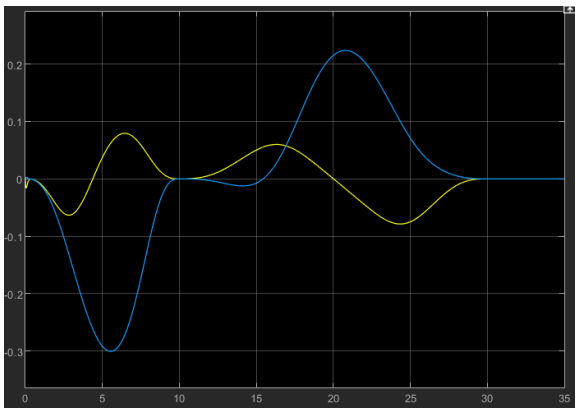


L'unica variazione rispetto alla compensazione perfetta è che nell'istante iniziale entrambe le coppie presentano una piccola variazione.

Se si osservano le variabili di giunto effettive calcolate dal blocco della dinamica:

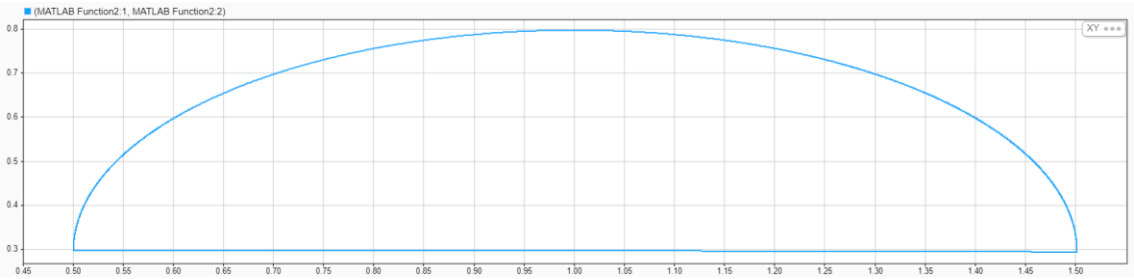


$\boldsymbol{q(t)}$



$\boldsymbol{\dot{q}(t)}$

Nello spazio di lavoro:



Anche in questo caso ci sono piccoli errori nel tratto rettilineo.

Controllore con compensazione delle forze apparenti

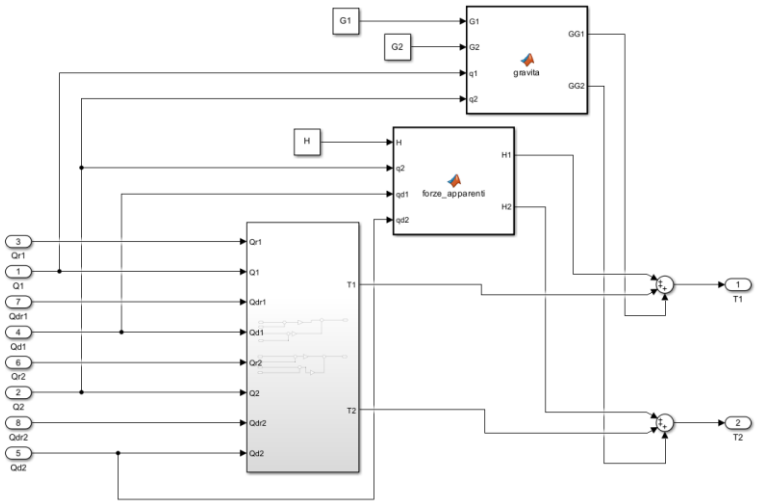
Compensazione perfetta

Il controllore in questione descrive:

$$T_i(t) = K_{P_i}\left[q^*{}_i(t) - q_i(t)\right] + K_{D_i}\left[q^*{}_i\dot{}(t) - q_i\dot{}(t)\right] + G(Q) + h(Q,\dot{Q})\dot{Q}$$

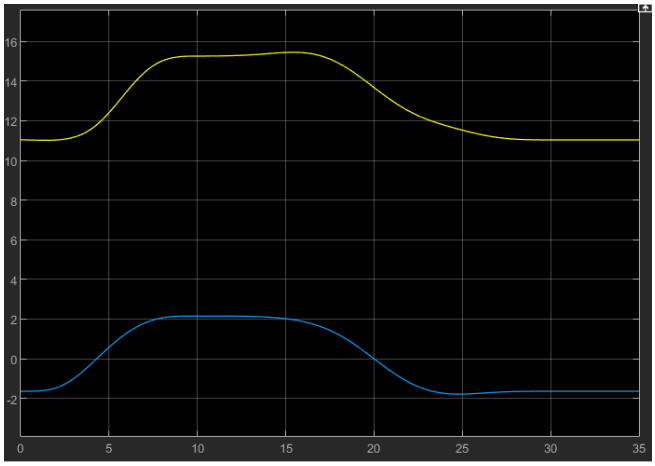
Quindi, in questo caso si compensa, oltre al vettore gravità, il vettore delle forze apparenti.

In *simulink*:



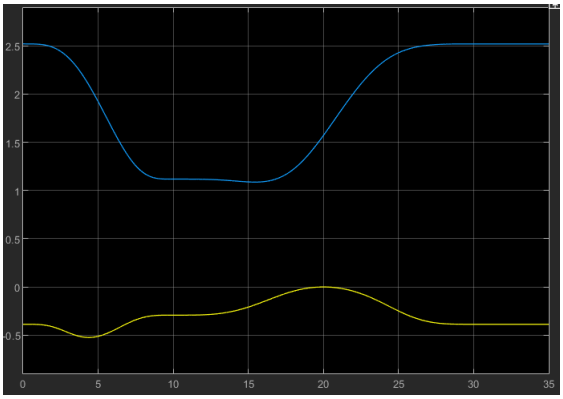
In questo caso, $\boldsymbol{K_P = 180}$ e $\boldsymbol{K_D = 17}$.

Se si osservano le coppie:

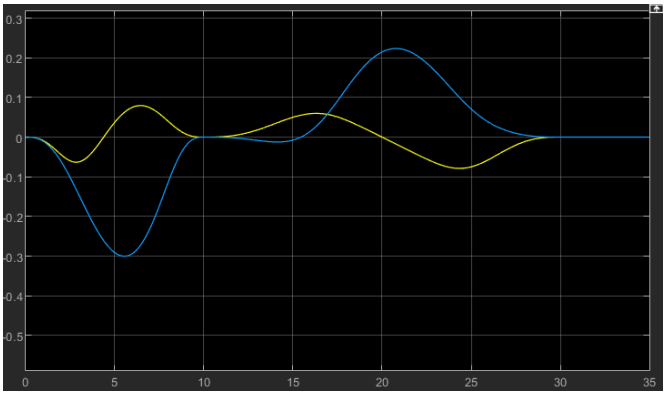


Come per il controllore che compensa perfettamente la gravità, quest'ultimo genera coppie che per $\boldsymbol{t \to \infty}$, le coppie tendono al loro valore iniziale.

Se si considerano le variabili di giunto effettive:

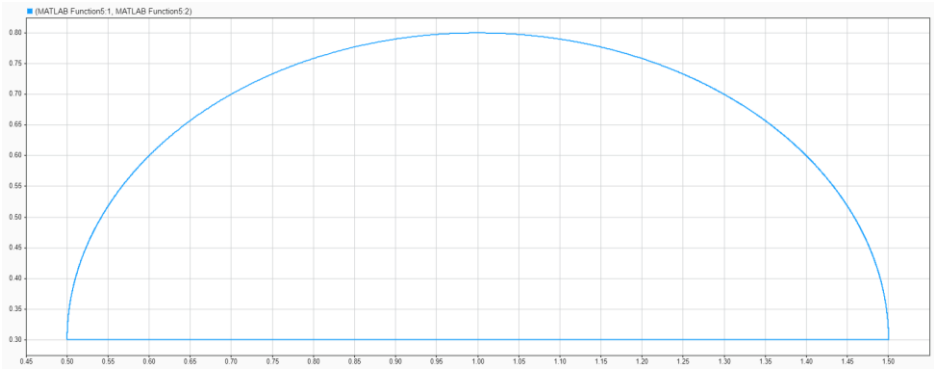


$\boldsymbol{q(t)}$



$\boldsymbol{\dot{q}(t)}$

Nello spazio di lavoro:



Non sono presenti errori nemmeno in questo caso proprio perché è dovuto alla compensazione perfetta della gravità e delle forze apparenti.

Compensazione non perfetta

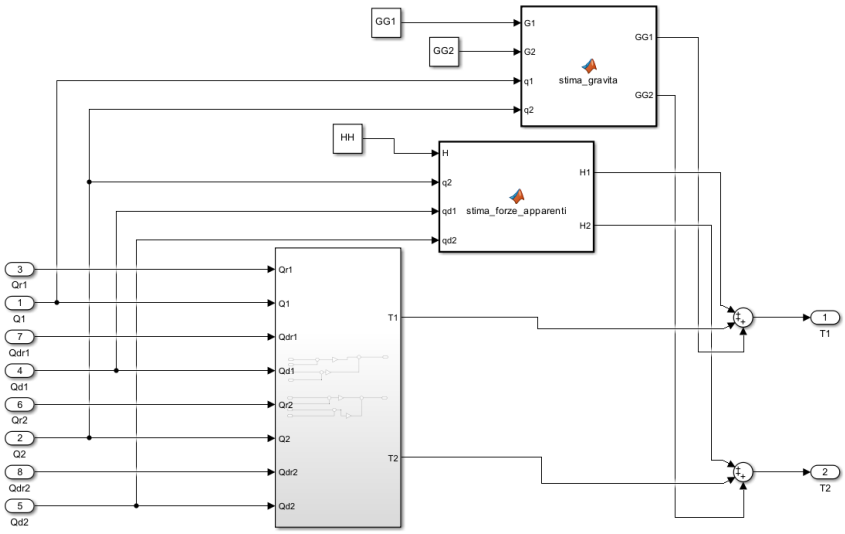
Il controllore, in questo caso, descrive:

$$T_i(t) = K_{P_i}\left[q^*{}_i(t) - q_i(t)\right] + K_{D_i}\left[q^*{}_i\dot{}(t) - q_i\dot{}(t)\right] + \hat{G}(Q) + \hat{h}(Q,\dot{Q})\dot{Q}$$

dove $\widehat{\boldsymbol{G}}(\boldsymbol{Q})$ e $\widehat{\boldsymbol{h}}(\boldsymbol{Q},\dot{\boldsymbol{Q}})\dot{\boldsymbol{Q}}$ sono stime del vettore gravità e del vettore delle forze apparenti.

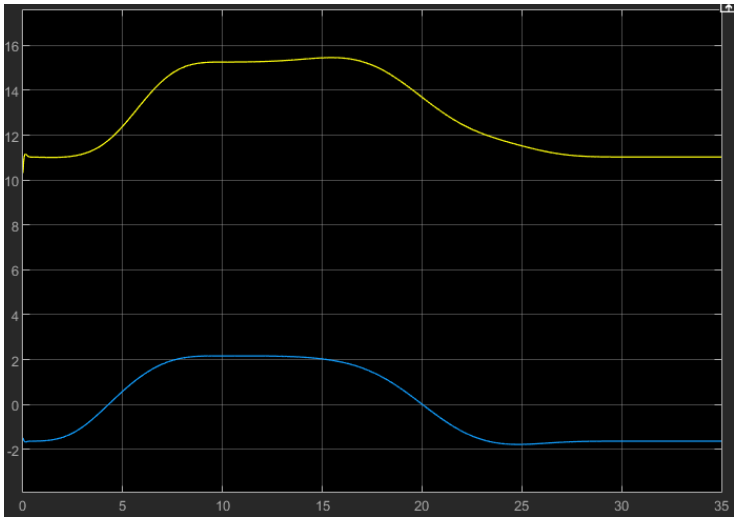
Anche in questo caso il controllore non conosce le masse perfette dei bracci ma conosce delle *masse fittizie*; si prendono in considerazione le masse citate in precedenza.

In *simulink*:



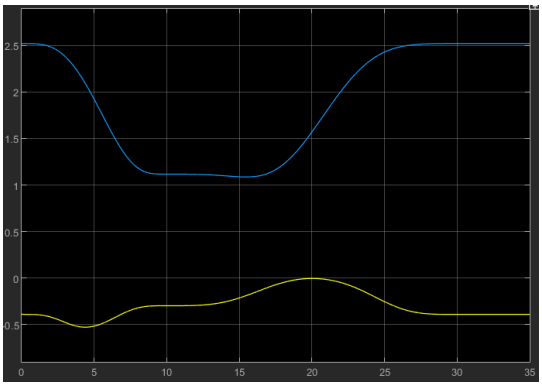
In questo caso, $\boldsymbol{K_P = 265}$ e $\boldsymbol{K_D = 25}$.

Il controllore genera le seguenti coppie:

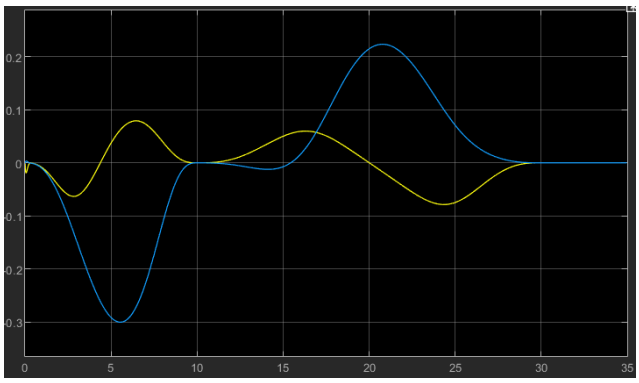


Si nota una piccola variazione all'istante iniziale dovuta alla conoscenza non perfetta delle masse.

Le coppie quindi, anche in questo caso, arrivano al blocco della dinamica e calcolano le variabili di giunto:

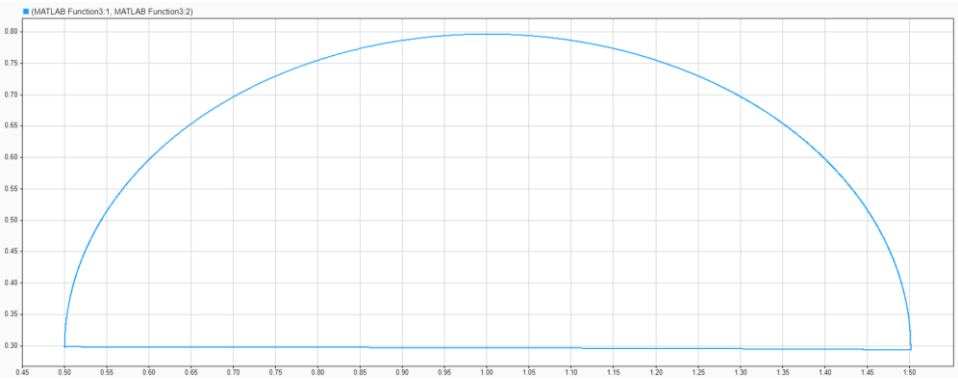


$\boldsymbol{q(t)}$



$\boldsymbol{\dot{q}(t)}$

Nello spazio di lavoro:



Anche in questo caso si notano piccoli errori durante il tratto rettilineo.