Risultato: 0

(a cura del docente)

Esame di: **Fondamenti di Robotica**
Consegna: **19 marzo 2024**

Gruppo: **FR22-05**
Studenti: **Catalano Gabriele, 0711967** (corris pondente)
Castello Giorgio, 0737589
Fiorentino Vincenzo Maria, 0650761
Li Pira Riccardo, 0714123

2 Parte Robotica

Una fabbrica impiega *bracci robotici* per inscatolare i prodotti provenienti da diverse linee di produzione, *muletti autonomi* per trasportare tali prodotti, e *nastri trasportatori* per traslare le scatole piene verso le linee di uscita. Precisamente, una sezione della fabbrica ospita un braccio articolato orizzontale, formato da due link di lunghezza 1.5 [m], ciascuno, e due giunti rotoidali; i prodotti arrivano da due linee di origine (LA e LB), che si trovano rispettivamente in posizione $(-10, 0)$ [m] e $(10, 0)$ [m]; i prodotti provenienti da LA devono essere trasportati da un primo muletto, M_1, nella posizione di "pick1" $(-3, 0)$ [m], mentre quelli provenienti da LB devono essere traslati da un secondo muletto, M_2, nella posizione di "pick2" in $(3, 0)$ [m]; il braccio robotico, alternativamente, prende un prodotto da una linea e lo inserisce in una scatola posizionata in posizione di "place" in $(0, 3)$ [m]; quando una scatola contiene prodotti da entrambe le linee, la scatola è considerata piena ed un nastro trasportatore la trasla verso una linea di uscita LC posizionata in $(0, 10)$ [m]. Questo meccanismo si ripete indefinitamente. Si chiede di progettare e simulare l'intero sistema sotto le seguenti ipotesi semplificative:

- la forza peso non ha nessun effetto sul braccio;
- la sincronizzazione tra il braccio e ciascun muletto avviene nel momento in cui la posizione dell'estremità del braccio e quella del muletto si trovano nella rispettiva posizione di "pick" a meno di una soglia di tolleranza di distanza di 0.010 [m];
- eventuali sovrapposizioni dei vari sottosistemi, che nella realtà genererebbero urti, si trascurano;
- il "buffer" delle posizioni di pick è nullo, perciò ciascun muletto, quando raggiunge la propria posizione di pick, aspetta il braccio robotico prima di tornare alla propria linea di origine;
- il nastro trasportatore con sopra il carico di una scatola si comporta come un sistema ad un polo dominante con funzione di trasferimento $G(s) = 1/(1 + s\tau)$, con $\tau = 0.9$ [s]; la traslazione si ferma non appena la scatola ha raggiunto LC;
- il braccio robotico aspetta che la scatola abbia raggiunto LC per partire con un nuovo pick and place;
- non ci sono requisiti sulla traiettoria del braccio nel movimento dalle posizioni di "pick" a quella di place, mentre è solo richiesto che l'estremità del braccio raggiunga tali punti;
- ciascun muletto è un uniciclo.

L'organizzazione della soluzione, nonché la sua presentazione, sono lasciate libere agli studenti. Tuttavia, affinché essa siano ritenute corrette e complete, devono descrivere chiaramente almeno:

- il modello dinamico di ciascun componente coinvolto (braccio, muletti, nastro, ...) e la tipologia di operazione che deve svolgere (pick-and-place, point-to-point, trajectory tracking, path following, ...);

- uno o più opportuni controllori di basso livello per ciascun componente e per ciascuna operazione che deve svolgere, nonché la descrizione della procedura teoria seguita per il suo ottenimento;
- i pianificatori e gli automi per la sincronizzazione delle varie fasi e la sincronizzazione dei vari componenti (M_1 con il braccio, M_2 con il braccio, braccio con il nastro, ...);
- Il risultato di una o più simulazioni, con i rispettivi diagrammi Matlab/Simulink dei vari sottosistemi e i rispettivi codici Matlab, ciascuno chiuso in anello singolo con il proprio controllore; le simulazione devono mostrare il raggiungimento dell'obiettivo, riportando almeno l'andamento dei segnali di interesse, di tutte le variabili di posizione, dei segnali di comando e di sincronizzazione.

Soluzione:

- **Braccio robotico**

  Per ottenere il modello dinamico del braccio secondo il metodo di Eulero-Lagrange, si procede seguendo i seguenti punti:

  1. Si fissa un sistema di riferimento SR_0 assoluto, in questo caso posto alla base con asse z lungo l'asse di rotazione del primo giunto
  2. si sceglie di un insieme di coordinate generalizzate $q = (q1,\, q2,\, \ldots\, ,\, qN)$ che descrivano la configurazione cinematica del sistema (sono le N coordinate di giunto)
  3. si calcola l'energia cinetica $T = T(q,\, \dot{q})$ e l'energia potenziale $U = U(q)$ dei corpi rigidi (nel nostro caso dato che la forza peso non ha effetto sul braccio $U = 0$)
  4. si costruisce il Lagrangiano $L(q,\, \dot{q}) = T(q,\, \dot{q}) - U(q)$
  5. si impone la seguente equazione di Eulero-Lagrange:

  $$\frac{d}{dt}\left(\frac{dL}{d\dot{q}}\right)^T - \left(\frac{dL}{dq}\right)^T = \tau$$

  Con τ coppie/forze non conservative che producono lavoro.

  Il modello dinamico risultante assume la seguente forma generale:

  $$M(q)\ddot{q} + c(q,\dot{q}) + e(q) = \tau$$

  Con $M(q)$ matrice di inerzia generalizzata;

  $c(q,\dot{q})$ vettore di forze/coppie centrifughe e di Coriolis;

  $e(q)$ vettore di forze/coppie dovute all'energia potenziale;

  Nel nostro caso:
  Considerando $N = 2$ giunti rotanti, q_1 posizione angolare del primo braccio rispetto l'asse x, q_2 posizione angolare relativa del secondo braccio rispetto al

primo braccio, l lunghezza braccio e d distanza del baricentro del braccio dal proprio asse di rotazione. Avremo le seguenti grandezze cinematiche:

$$p_c1 = \begin{pmatrix} dcos(q_1) \\ dsin(q_1) \\ 0 \end{pmatrix} \rightarrow v_c1 \begin{pmatrix} -dsin(q_1)(\dot{q}_1) \\ -dcos(q_1)(\dot{q}_1) \\ 0 \end{pmatrix}$$

$$p_c2 = \begin{pmatrix} lcos(q_1) + dcos(q_1 + q_2) \\ lcos(q_1) + dsin(q_1 + q_2) \\ 0 \end{pmatrix} \rightarrow v_c2 \begin{pmatrix} -(lsin(q_1)(\dot{q}_1) + dsin(q_1 + q_2)sin(\dot{q}_1 + \dot{q}_2)) \\ lcos(q_1)(\dot{q}_1) + dcos(q_1 + q_2)(\dot{q}_1 + \dot{q}_2) \\ 0 \end{pmatrix}$$

$$\omega_1 = \begin{pmatrix} 0 & 0 & \dot{q} \end{pmatrix}^T \qquad , \qquad \omega_2 = \begin{pmatrix} 0 & 0 & \dot{q}_1 + \dot{q}_2 \end{pmatrix}^T$$

L'energia cinetica:

$$T_i = \frac{1}{2} m_i (v_{c_i})^T (v_{c_i}) + \frac{1}{2} (\omega_i)^T I_i \omega_i \qquad \text{dei due bracci } (T = T_1 + T_2)$$

$$T_1 = \frac{1}{2} m d \dot{q}_1^2 + \frac{1}{2} I_{zz} \dot{q}_1^2$$

$$T_2 = \frac{1}{2} m [l^2 \dot{q}_1^2 + d^2 (\dot{q}_1 + \dot{q}_2)^2 + 2ldcos(q_2) \dot{q}_1 (\dot{q}_1 + \dot{q}_2)] + \frac{1}{2} I_{zz} (\dot{q}_1 + \dot{q}_2)^2$$

con m massa del braccio e I_z inerzia.

A questo punto, costruendo il Lagrangiano $L = T - U$ ed eseguendo le derivate, si ottengono 2 equazioni differenzilali non lineari del 2° ordine, che descrivono il modello dinamico del robot. Sfruttando però le proprietà generali del modello si possono ricavare gli elementi di $M(q)$ e $c(q, \dot{q})$ dalla $T = T_1 + T_2$ individuando le funzione di q che pesano i prodotti $\dot{q}_i$, $\dot{q}_j$ per $i, j \in \{1, 2\}$, si ricavano gli elementi della matrice di inerzia:

$$M(q) = \begin{pmatrix} a_1 + 2a_2c_2 & a_3 + a_2c_2 \\ a_3 + a_2c_2 & a_3 \end{pmatrix}$$

Posto per compattezza $c_2 = cos(q_2)$ e $a_1 = I_{zz} + md^2 + I_{zz} + m(l^2 + d^2) > 0$, $a_2 = mld > 0$, $a_3 = I_{zz} + md^2 > 0$.

Dato che ciascuna componente $c_i (i = 1, ..., N)$ del vettore di forze/coppie centrifughe e di Coriolis è una forma quadratica nelle velocità $\dot{q}$:

$$c_i(q, \dot{q}) = \dot{q}^T C_i \dot{q}$$

Con la matrice C_i ottenuta per derivazione di elementi della $M(q)$:

$$C_i = \frac{1}{2} \left[\left(\frac{\partial M_i}{\partial q} \right) + \left(\frac{\partial M_i}{\partial q} \right)^T - \frac{\partial M_i}{\partial q} \right]$$ si ottiene (con $s_2 = \sin q_2$):

$$C_1(q) = \begin{pmatrix} 0 & -a_2 s_2 \\ -a_2 s_2 & -a_2 s_2 \end{pmatrix} \quad , \quad C_2(q) = \begin{pmatrix} a_2 s_2 & 0 \\ 0 & 0 \end{pmatrix}.$$

Da cui:

$$c(q,\, \dot{q}) = \begin{pmatrix} a_2 s_2 \dot{q}_2(\dot{q}_2 + 2\dot{q}_1) \\ a_2 s_2 \dot{q}_1^2 \end{pmatrix}$$

Riassumendo, in forma compatta il modello dinamico del robot è:

$$\underbrace{\begin{pmatrix} a_1 + 2a_2c_2 & a_3 + a_2c_2 \\ a_3 + a_2c_2 & a_3 \end{pmatrix}}_{M(q)>0} \cdot \begin{pmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{pmatrix} + \underbrace{\begin{pmatrix} a_2 s_2 \dot{q}_2(\dot{q}_2 + 2\dot{q}_1) \\ a_2 s_2 \dot{q}_1^2 \end{pmatrix}}_{c(q,\, \dot{q})} = 0$$

Per il controllore si è utilizzato l'approccio della linearizzazione esatta del modello dinamico: $u = M(q)\ddot{q} + c(q,\, \dot{q})$, ponendo $\ddot{q} = K_p(q_d - q) - K_d\dot{q}$ con q_d stato assegnato e K_p, K_d costanti positive. Il controllore complessivo è non lineare $u = M(q)[K_p(q_d - q) - K_d\dot{q}] + c(q,\, \dot{q})$, nel nostro caso ponendo $\ddot{q} = \ddot{q}_d + K_d(\dot{q}_d - q)$ il controllore risulta contenere termini di feedforward e feedback:

$$u = M(q)[\ddot{q}_d + K_d(\dot{q}_d - \dot{q}) + K_p(q_d - q)] + c(q,\, \dot{q})\,.$$

- **Muletto**

Utilizziamo il modello dinamico dell'uniciclo trasformandolo in coordinate polari:

$$\begin{cases} \dot{x} = vcos\theta \\ y = vsin\theta \\ \dot{\theta} = \omega \\ \dot{v} = \frac{1}{m}\tau_v \\ \dot{\omega} = \frac{1}{I_z}\tau_\theta \end{cases} \rightarrow \begin{cases} \rho = \sqrt{(x - X_i)^2 + (y - Y_i)^2} \\ \phi = arctan((y - Y_i)/(x - X_i)) \\ \beta = arctan((y - Y_i)/(x - X_i)) + \pi - \theta \end{cases} \rightarrow \begin{cases} \dot{\rho} = -\rho cos\beta\bar{v} \\ \dot{\phi} = sin\beta\bar{v} \\ \dot{\beta} = sin\beta\bar{v} - \omega \\ \dot{\bar{v}} = \frac{\tau_v}{m\rho} + cos\beta\bar{v}^2 \\ \dot{\omega} = \frac{\tau_\omega}{I_z} \end{cases}$$

Implementiamo il controllo di postura:

La candidata di Lyapunov utilizzata e la relativa derivata direzionale sono riportate di seguito:

$$\begin{cases} V = \frac{1}{2}\rho^2 + \frac{1}{2}\phi^2 + \frac{1}{2}\beta^2 \\ \dot{V} = -\rho^2 cos\beta\bar{v} + \phi sin\beta\bar{v} + \beta sin\beta\bar{v} - \beta\omega \end{cases}$$

Dal controllo cinematico otteniamo il seguente controllore:

$$\begin{cases} \bar{v} = \Gamma_{\bar{v}} = cos(\beta) \\ \omega = \Gamma_\omega = (\phi + \beta)sinc(\beta)cos(\beta) + \beta \end{cases}$$

Implementiamo il controllo sugli ingressi $\tau_{\dot{v}}$ e τ_ω utilizzando il backstepping, definiamo la candidata di Lyapunov estesa:

$$W = V + \frac{1}{2}(\bar{v} - \Gamma_{\bar{v}})^2 + \frac{1}{2}(\omega - \gamma_\omega)^2$$

E la conseguente derivata di Lyapunov estesa:

$$\dot{W} = \dot{V} + (\bar{v} - \Gamma_{\bar{v}})(\frac{\tau_v}{m\rho} + cos\beta\bar{v}^2 - \dot{\Gamma}_{\bar{v}}) + (\omega - \Gamma_\omega)(\frac{\tau_\omega}{I_z} - \dot{\Gamma}_\omega)$$

Dunque otteniamo il seguente controllore per il modello dinamico:

$$\begin{cases} \tau_{\dot{v}} = m\rho(\dot{\Gamma}_{\bar{v}} - cos\beta\bar{v}^2 - K_{bv}(\bar{v} - \Gamma_{\bar{v}}) \\ \tau_\omega = I_z(\dot{\Gamma}_\omega - K_{b\omega}(\omega - \Gamma_\omega)) \end{cases}$$

- **Nastro**

  Viene implementata la funzione di trasferimento: $G(s) = \frac{1}{(1 + s\tau)}$ con $\tau = 9$ e condizione d'arresto in LC .

Per quanto concerne la fase di progettazione e simulazione del sistema di imballaggio preso in esame, abbiamo dunque deciso di optare per la scelta di un sistema di pick-and-place per il braccio robotico, point to point per il controllo dei due muletti M_1 e M_1 ed infine nei riguardi del nastro implementato la funzione di trasferimento tramite block parameter.

Settiamo le condizioni iniziali tramite codice Matlab runme.m :

```
clearvars, close all, clc

%variabili muletti
M1_Ref = [3 0; 10 0];
M2_Ref = [-3 0; -10 0];
M1_State0 = [10; 0; 0; 0; 0];
M2_State0 = [ -10; 0; 0; 0; 0];

M_m = 2;
M_I = 1;
M_kbv = 1;
M_kbomega = 1;

M_constants = [M_m M_I M_kbv M_kbomega];

%variabili Braccio
B_Ref = [0 180; 90 180; 180 180];
B_State0 = [0; 0];

l = 1.5;
db = 0.75;
m = 0.8;
I = 0.71;

B_constants = [l, db, m, I];

%variabili nastro
T = 0.9;
RefVector = [3 10];
```

Per sincronizzare l'intero progetto e avere un corretto funzionamento delle varie componenti del sistema abbiamo deciso di sfruttare un unico planner globale (di cui in figura 9 la rappresentazione del diagramma Simulink appositamente creato) implementato con il seguente codice Matlab:

```
function [M1_r,M2_r,B_r,N_r]=Globalplanner(M1_q,M2_q,B_q,N_q,M1_Ref,M2_Ref,B_Ref,N_Ref)
    TOLERANCE = 0.01;
     persistent M1_p;
     if isempty(M1_p); M1_p = 1; end
     persistent M2_p;
     if isempty(M2_p); M2_p = 1; end
     persistent B_p;
     if isempty(B_p), B_p = 1; end
     persistent N_p;
     if isempty(N_p), N_p = 1; end

    B_tasks = [1,2,3,2]; % array delle task da eseguire del braccio

    %M1 Operation
    M1_rho = M1_q(1);
    M1_r = M1_Ref(M1_p,:);
    M1_xd = M1_r(1);

    %M2 Operation
    M2_rho = M2_q(1);
    M2_r = M2_Ref(M2_p,:);
    M2_xd = M2_r(1);


    %B operation
    B_q1 = B_q(1);
    B_q2 = B_q(2);
    B_r = B_Ref(B_tasks(B_p),:)';

    %N operation
    N_r = N_Ref(N_p);

    %condizioni di raggiungibilita'
    M1_rp=abs(M1_rho-M1_xd) < TOLERANCE; %M1 ha raggiunto il waypoint
    M2_rp=abs(M2_rho-M2_xd) < TOLERANCE; %M2 ha raggiunto il waypoint
    B_rp=abs(B_q1-B_r(1)) + abs(B_q2-B_r(2)) < TOLERANCE;%B ha raggiunto il waypoint
    N_rp=abs( N_q-N_r) < TOLERANCE; %N ha raggiunto il waypoint
    B_f=(B_p==4) && B_rp;%B ha terminato il loop

     if M1_p == 2 && M1_rp, M1_p = 1; return; end %M1 ritorna in posizione LB (-10,0)
     if M2_p == 2 && M2_rp, M2_p = 1; return; end %M2 ritorna in posizione LA (10,0)

     if N_rp && N_p == 2, N_p = 1; end %N ritorna in posizione di place (0, 3)


    %M1 ha raggiunto la posizione pick1 (-3,0) e M2 ha raggiunto la
    %posizione di pick2 (3,0)
     if M1_p == 1 && M1_rp && M2_p == 1 && M2_rp

       %N inizia a raggiungere LC (0,10) quando il B ha terminato la posizione di place
         if B_rp && B_p == 4
             N_p = 2;
         end
         %loop B -> pick1, place, pick2, place
         if (B_rp && not(B_f))
              if B_p == 1,B_p=2; elseif B_p==2,B_p=3; elseif B_p==3,B_p=4; end
         end

         %B ha completato il loop, M1 inizia a raggiungere la posizione di
         %LB e M2 inizia a raggiungere la posizione di LA, B inizia a
         %raggiungere la posizione di pick1
         if B_f
             B_p = 1;
             M1_p = 2;
             M2_p = 2;
         end
    end
end
```

Il braccio robotico (diagramma Simulink in figura 8) viene implementato tramite funzione Matlab:

dynamics:

```
function ddq = dynamics(dq, q, u, constants)

%unpacks input
q1 = q(1); q2 = q(2);
dq1 = dq(1); dq2 = dq(2);
l = constants(1);
d = constants(2);
m = constants(3);
I = constants(4);



%dynamics
a1 = I + m*d^2 + I + m*(l^2 + d^2);
a2 = m*l*d;
a3 = I + m*d^2;

B = [a1 + 2*a2*cos(q2),a3 + a2*cos(q2);a3 + a2*cos(q2), a3];
C = [a2*sin(q2)*dq2*(dq2 + 2*dq1);a2*sin(q2)*dq1^2];

ddq = B \ (u - C);

end
```

controller:

```
function u = controller(q, dq, ref, dref, ddref, constants)

%unpacks input
q1 = q(1); q2 = q(2);
dq1 = dq(1); dq2 = dq(2);

lambda = 5;
Kp = lambda*2;
Kd = lambda^2;

l = constants(1);
d = constants(2);
m = constants(3);
I = constants(4);


%dynamics
a1 = I + m*d^2 + I + m*(l^2 + d^2);
a2 = m*l*d;
a3 = I + m*d^2;

B = [a1+2*a2*cos(q2),a3+a2*cos(q2);a3+a2*cos(q2),a3]; %matrice di inerzia generalizata
C = [a2*sin(q2)*dq2*(dq2 + 2*dq1);a2*sin(q2)*dq1^2];



u = B*[ddref + Kd*(dref - dq) + Kp*(ref - q)] + C;

end
```

I due muletti (diagramma Simulink in figura 6 e 7) vengono implementati tramite funzione Matlab:

dinamica uniciclo:

```
function dState = unidinam(State, u, constants, kitematics, Ref)
    %unpack inputs
    rho = State(1) - Ref(1);
    phi = State(2);
    beta = State(3);


    tauV = u(1);
```

```
    tauOmega = u(2);


    Tv = kitematics(1);
    Tomega = kitematics(2);

    m = constants(1);
    I = constants(2);

    %pack outputs
    dState = [-rho*cos(beta) * Tv ;
        sin(beta) * Tv;
        sin(beta) * Tv - Tomega;
        tauV / (m * rho) + cos(beta) * Tv^2;
        tauOmega / I];
end
```

dinamica point-to-point uniciclo:

```
function tau = unip2pctrldinam(State, T, dT, constants, Ref)
    %unpack inputs
    rho = State(1)-Ref(1);
    beta = State(3);

    v = State(4);
    omega = State(5);

    Tv = T(1);
    Tomega = T(2);

    dTv = dT(1);
    dTomega = dT(2);

    m = constants(1);
    I = constants(2);
    kbv = constants(3);
    kbo = constants(4);

    tauv= m*rho*(dTv-cos(beta*v^2)-kbv*(v-Tv));
    tauo= I*(dTomega-kbo*(omega-Tomega));

    %pack outputs
    tau = [tauv; tauo];
end
```

controller point-to-point uniciclo

```
function T = unip2pctrl(State)
    phi = State(2);
    beta = State(3);
    v = cos(beta);
    omega = (phi + beta)*sinc(beta) * cos(beta) + beta;
    T = [v; omega];
end
```

Tramite la figura 10 e 11 possiamo visionare l'andamento dei segnali esaminati e quindi la simulazione del corretto funzionamento con una breve descrizione passo passo della successione contemporanea degli eventi che caratterizzano il sistema di imballaggio pacchi creato.

Figure e diagrammi

1 - PARTE CONTROLLO NON LINEARE

parte controllo lineare punto d

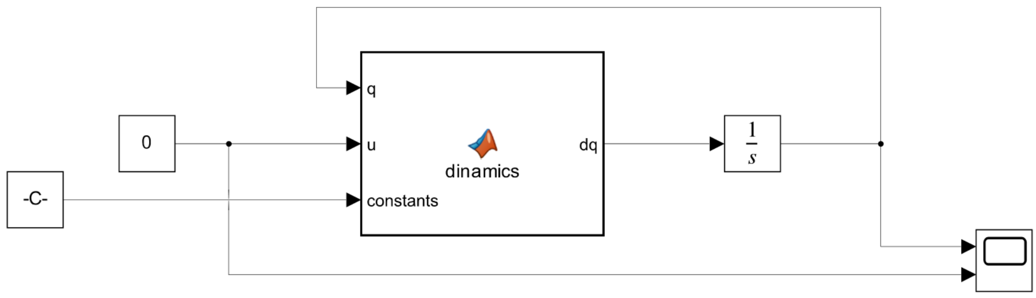

Figura 1: Diagramma Simulink del sistema controllato .

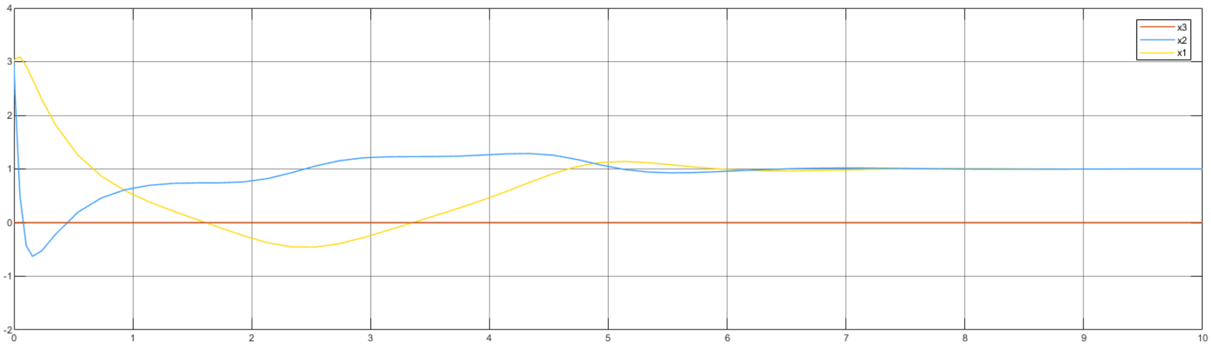

Figura 2: Andamento delle variabili esaminate .

parte controllo lineare punto f

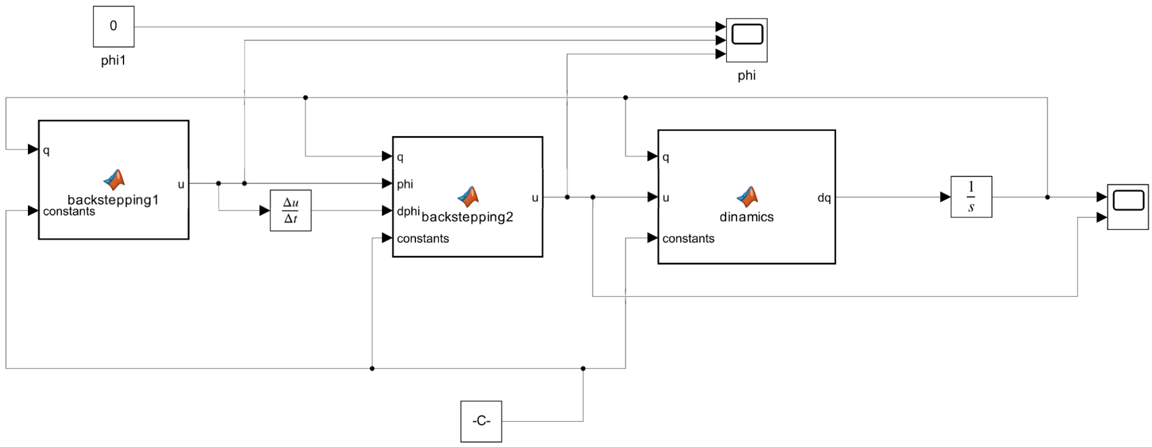

Figura 3: Diagramma Simulink del sistema controllato con uso di Backstepping.

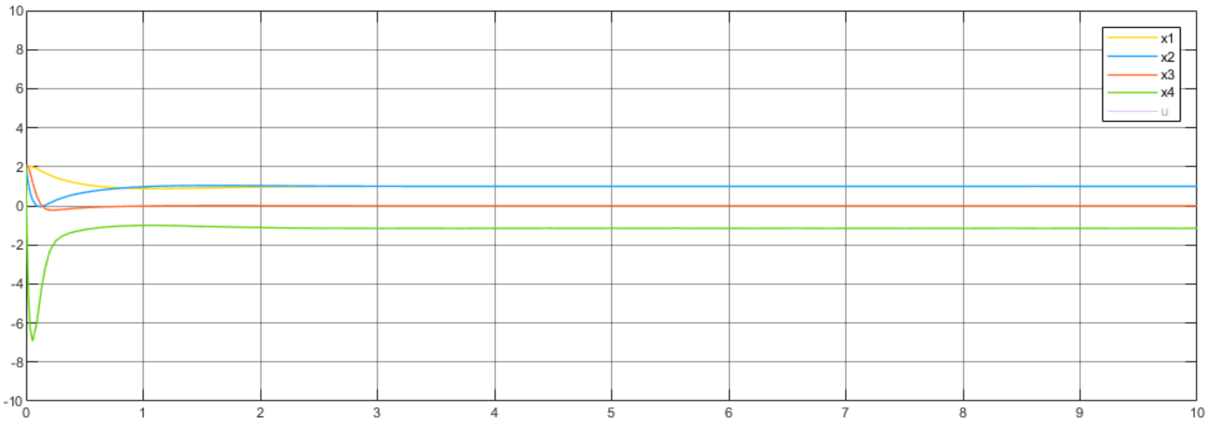

Figura 4: Andamento delle variabili di stato del sistema controllato.

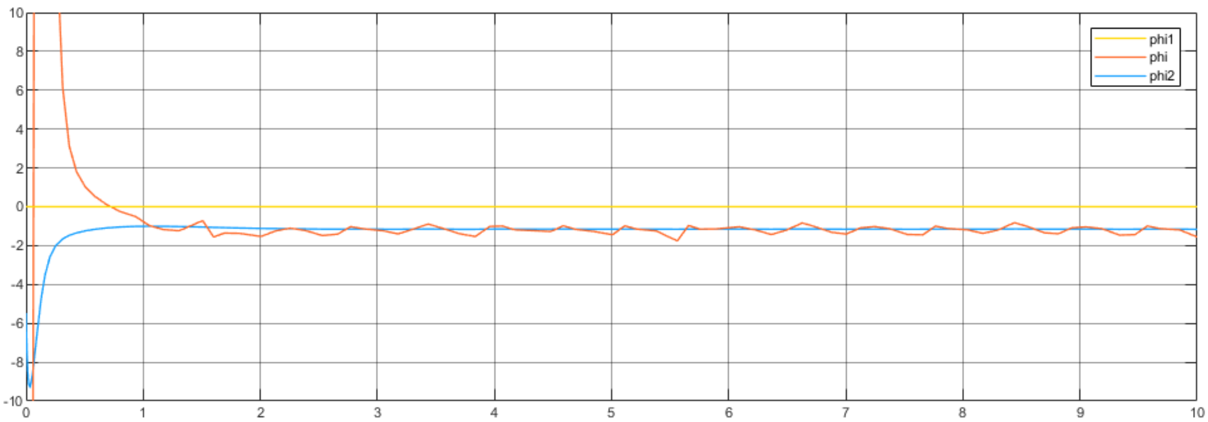

Figura 5: Andamento dei segnali di φ_1, φ_2, φ .

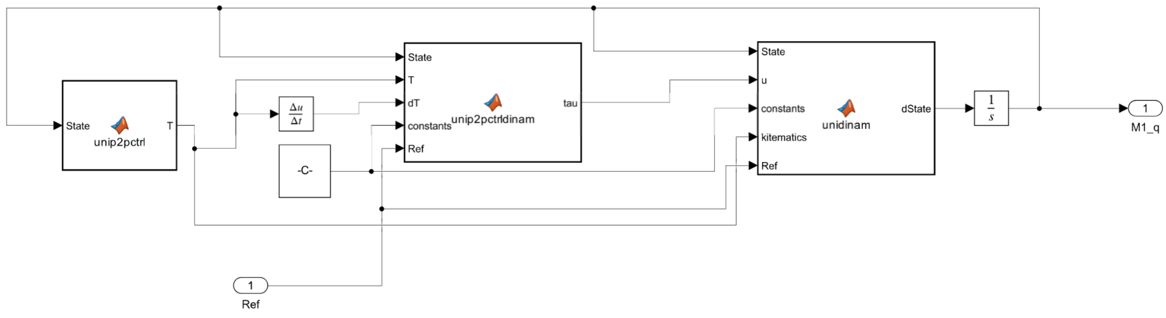

Figura 6: Diagramma Simulink del controllore relativo al muletto1

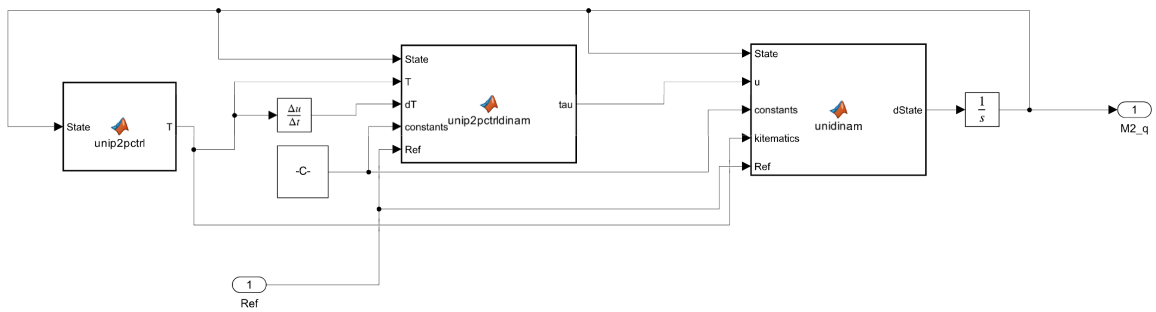

Figura 7: Diagramma Simulink del controllore relativo al muletto2

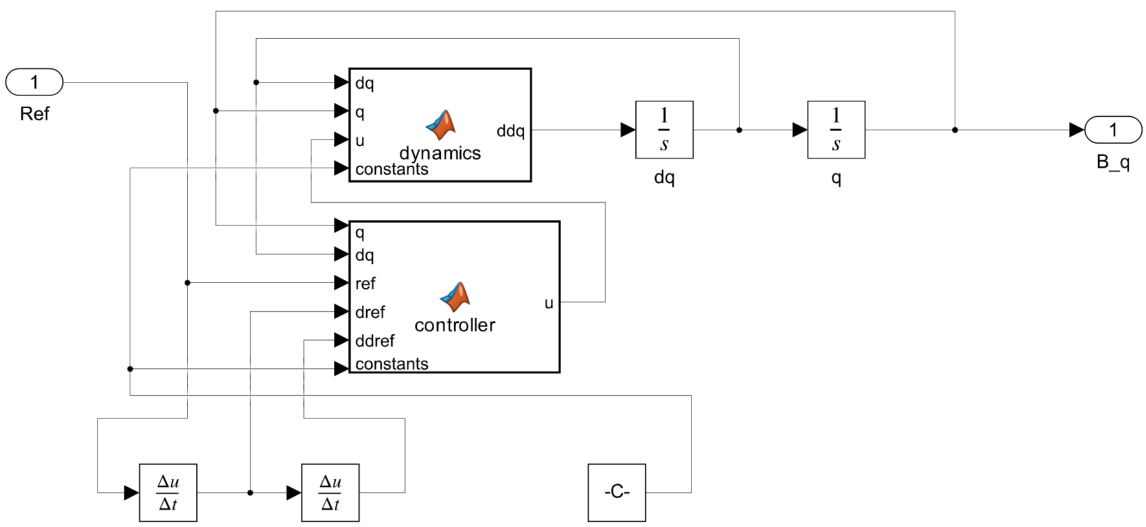

Figura 8: Diagramma Simulink del controllore relativo al braccio robotico

Fine dello svolgimento

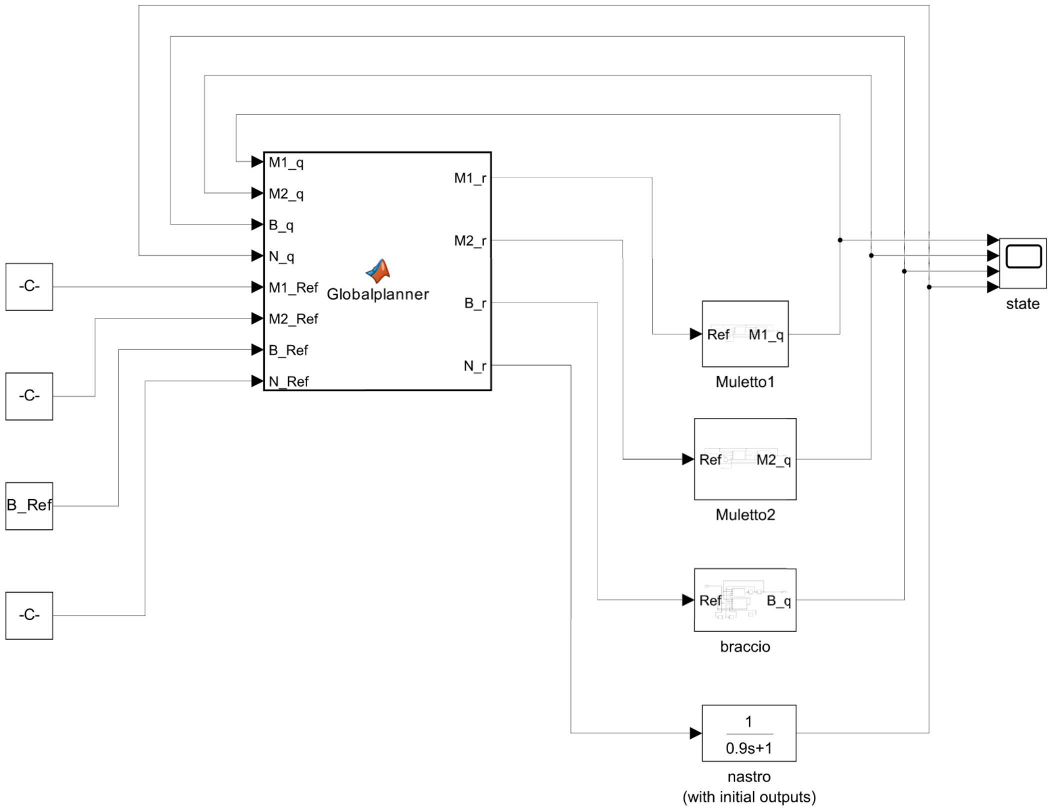

Figura 9: Diagramma Simulink del controllore globale dell'intero sistema di imballaggio

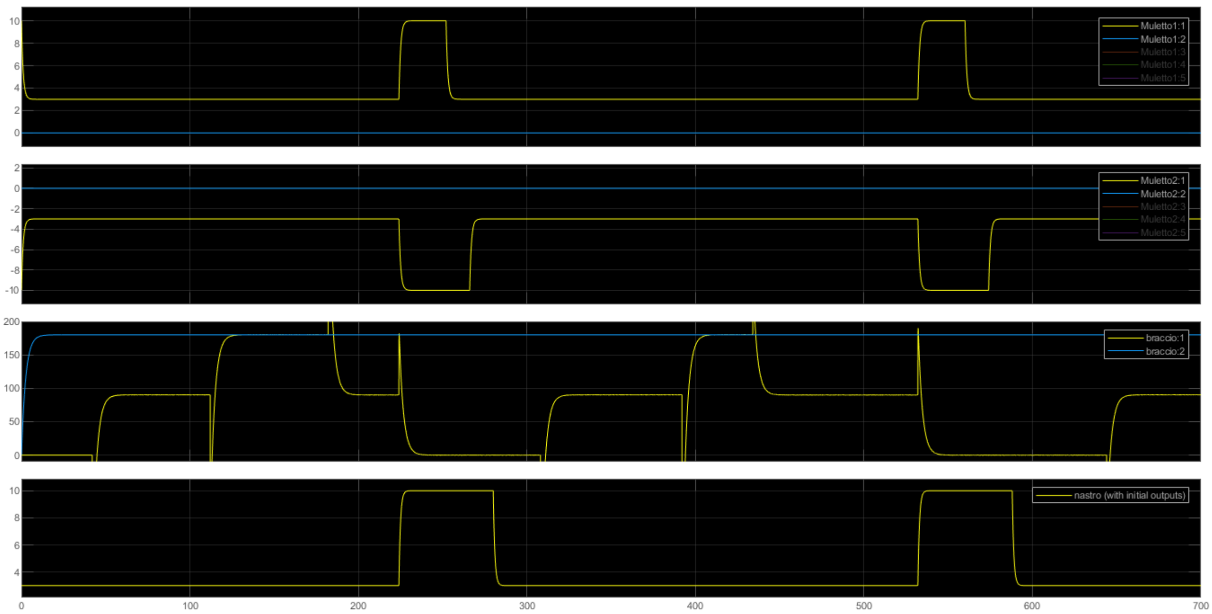

Figura 10: Andamenti del sistema controllato

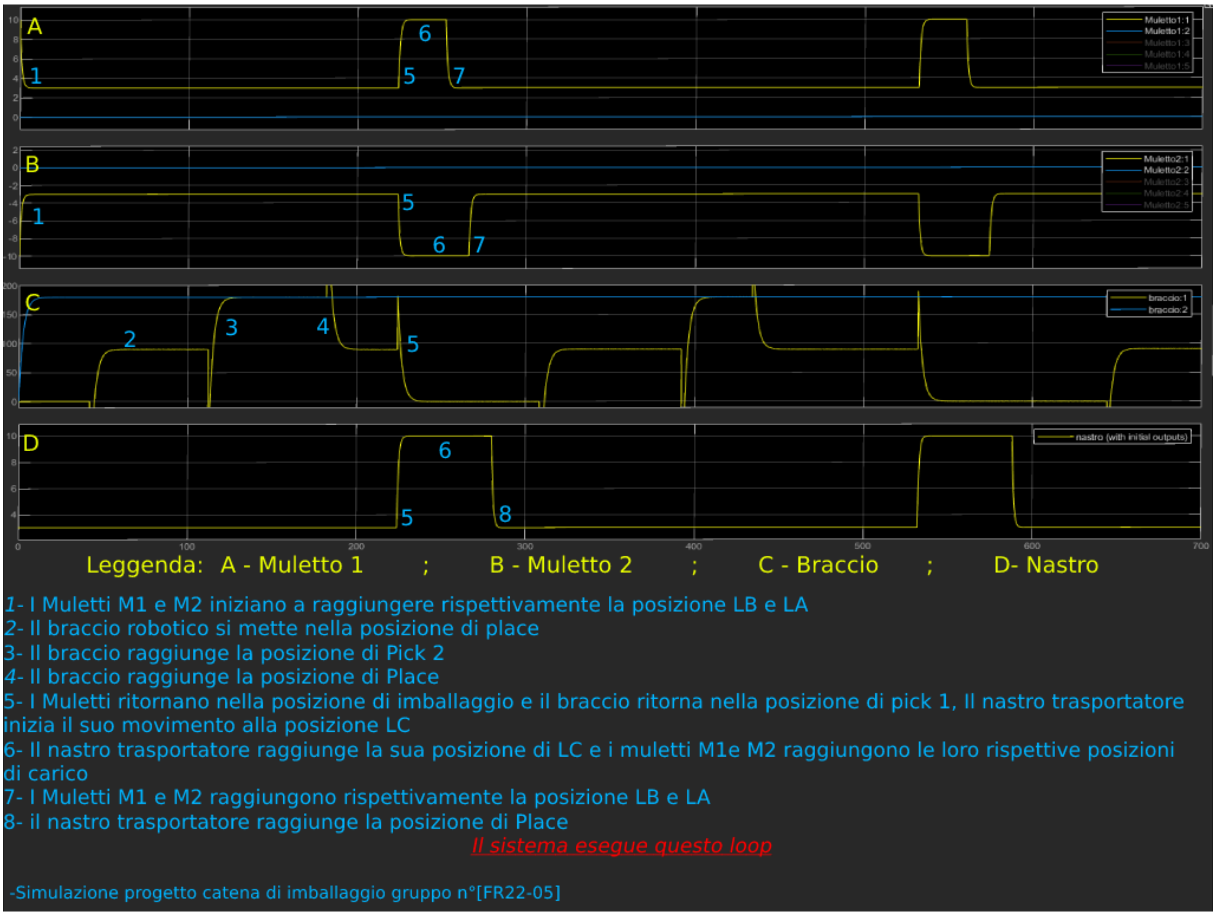

Figura 11: Descrizione delle fasi del processo di imballaggio