Università degli Studi Mediterranea di Reggio Calabria

Corso di Laurea Magistrale in Ingegneria Elettronica

"Controllo avanzato e multivariabile"

"Sintesi di un controllo per il dispositivo 'Hoverboard'"

Antonella Pulitanò matr 86092

1.Introduzione

L'**hoverboard**, un monopattino o monociclo elettrico autobilanciante privo di manubrio. È composto da una tavola con due piccole pedane, che si muovono sullo stesso asse e su cui poggiano i piedi del conducente, e con alle estremità due ruote che consentono una mobilità molto agevole su diversi tipi di terreno.

Per farlo funzionare, e anche per accelerare, è sufficiente mettere entrambi i piedi sopra le due pedane che compongono la tavola e **inclinarsi in avanti**. L'hoverboard dispone infatti di un sensore di peso che permette di farlo partire, mentre **per decelerare è necessario inclinarsi all'indietro** e per curvare a destra o a sinistra occorre fare una leggera pressione con i rispettivi piedi.

La sua **velocità massima** può variare molto a seconda del modello e del peso della persona che utilizza il dispositivo, ad esempio, più il conducente sarà pesante, più la velocità sarà limitata.
La maggior parte degli apparecchi che si trovano in commercio sono in grado di raggiungere una velocità massima **tra i 10 e i 20 km/h**, riesce a sostenere **fino a 100-120 kg** e ha un peso che varia **tra i 10 e i 12 kg.** La velocità del prodotto dipende, oltre che dal peso dell'utente, anche dalla **potenza dei due motori** di cui è dotato che, in genere, varia **tra 250 e 380 Watt**.

L'obiettivo di questo progetto consiste nella simulazione in ambiente matlab del funzionamento di questo dispositivo; ciò, nello specifico, prevede la progettazione di un regolatore che mantenga in equilibrio il sistema attraverso un'opportuna azione di controllo in termini di coppia da applicare alle ruote dell'hoverboard.

In più viene trattata anche una parte del controllo vincolato, ma solo dal punto di vista implementativo.

Il principio di funzionamento di questo dispositivo può essere associato a quello di un pendolo inverso, poiché esso bilancia l'inclinazione della persona, impressa con i piedi, con una velocità di traslazione.

Il pendolo inverso, infatti, è costituito da un carrello con collegato al suo centro di gravità un pendolo libero di oscillare; la parte più bassa può dunque muoversi per bilanciare le oscillazioni del pendolo e garantire così l'equilibrio.
Il problema di controllo consiste nel voler stabilizzare il sistema nel punto di equilibrio instabile, ovvero quando il pendolo è rivolto verso l'alto.

L'azione di controllo è quindi la forza impressa dal motore che permette al dispositivo di andare avanti o indietro bilanciare il pendolo attorno alla sua posizione instabile.

Il modello matematico del sistema è ricavato dall'analisi del sistema di forze, definito in base alla seconda legge di Newton, agenti sia sul carrello che sul pendolo; si ottiene così un sistema di due equazioni.

Si ottengono così le equazioni di stato che descrivono il sistema non lineare:

$$\begin{cases} (I + ml^2)\ddot{\theta} = -mlx^2 \cos\theta - mgl \sin\theta \\ (M + m)\ddot{x} = F - b\dot{x} - ml\ddot{\theta} \cos\theta + ml\dot{\theta}^2 \sin\theta \end{cases}$$

in cui si sono definite come variabili di stato: la posizione del carrello, x, la sua velocità, $\dot{x}$, l'angolo di inclinazione del pendolo, θ, rispetto alla verticale, la velocità angolare, $\dot{\theta}$.

Il sistema ha, dunque, quattro stati rappresentati dal vettore degli stati $[x\ \dot{x}\ \theta\ \dot{\theta}]^T$, un ingresso F(t), che è la forza applicata al carrello dall'attuatore e l'uscita coincide con lo stato.

2.Caratteristiche del sistema

2.1 Punti di equilibrio

Considerando un sistema tempo continuo $\dot{x} = f(x(t), u(t))$, x_{eq} è un punto di equilibrio per il sistema, con ingresso di equilibrio u_{eq}, se

$$f(x_{eq}, u_{eq}) = 0$$

Il pendolo inverso presenta due punti di equilibrio: uno si ha quando il pendolo si trova rivolto verso il basso, l'altro quando si trova rivolto verso l'alto.

Poiché la funzione che descrive il sistema nelle condizioni di equilibrio deve essere zero, annullando le derivate delle variabili di stato e risolvendo il sistema si ottengono i punti di equilibrio:

$$x_{eq1}[\nabla x\ 0\ 0\ 0]^T$$
$$x_{eq2}[\nabla x\ 0\ \pi\ 0]^T$$

Inoltre, in condizioni di equilibrio la forza applicata al carrello è nulla, $F_{eq} = 0$.

2.2 Verifica della stabilità dei punti di equilibrio

L'analisi della stabilità per un sistema non lineare viene basata sulla teoria di Lyapunov.

Nello specifico, si è fatto riferimento al metodo indiretto di Lyapunov, secondo il quale l'analisi della stabilità di un punto di equilibrio x_{eq} viene ricondotta allo studio della stabilità del corrispondente sistema linearizzato nell'intorno del punto di equilibrio considerato.

Infatti, se le funzioni che rappresentano il sistema risultano derivabili rispetto alle variabili di stato è possibile scrivere lo sviluppo in serie di Taylor di tali funzioni, considerando solo i termini del primo ordine, nell'intorno del punto di equilibrio.

È possibile così definire la matrice Jacobiana del sistema rispetto agli stati, J_x, composta dalle derivate parziali dell'equazioni di stato rispetto alle variabili di stato calcolate nel punto di equilibrio all'ingresso di equilibrio corrispondente.

$$J_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} \dots \dots \frac{\partial f_1}{\partial x_n} \\ \vdots \quad\quad\quad \vdots \\ \frac{\partial f_n}{\partial x_1} \dots \dots \frac{\partial f_n}{\partial x_n} \end{bmatrix}_{x_{eq}, u_{eq}}$$

Dato un punto di equilibrio, secondo il teorema di Lyapunov via linearizzazione, esso sarà;

- Asintoticamente stabile, per il sistema non lineare, se gli autovalori della matrice Jacobiana J_x sono tutti a parte reale negativa, e quindi se il sistema linearizzato è, a sua volta, asintoticamente stabile.
- Instabile, per il sistema non lineare, se vi sono alcuni autovalori della matrice Jacobiana J_x a parte reale positiva, e quindi se il sistema linearizzato è, a sua volta instabile.

Se, invece, la matrice Jacobiana J_x presenta autovalori a parte reale negativa e almeno un autovalore a parte reale nulla, allora la linearizzazione non fornisce informazioni sulla stabilità nell'intorno del punto di equilibrio.

Per applicare il metodo indiretto di Lyapunov, bisogna determinare la matrice Jacobiana dello stato valutata nel punto di equilibrio ad ingresso nullo. Essendo la matrice che si ottiene tramite la linearizzazione, la calcoliamo in ambiente matlab tramite il comando **linmod**.

La condizione di nostro interesse è definita dal punto di equilibrio $x_{eq2} = [0\ 0\ \pi\ 0]^T$. La matrice Jacobiana così ottenuta, presenta autovalori a parte reale positiva perciò il sistema linearizzato è instabile e, di conseguenza, x_{eq2} è un punto di equilibrio instabile per il sistema non lineare.

2.3 Raggiungibilità e osservabilità

Per progettare un controllo che possa stabilizzare il sistema in una qualsiasi condizione, cioè che possa far lavorare il sistema nella condizione di funzionamento di riferimento e importante che il sistema che si vuole regolare sia completamente raggiungibile (quindi, controllabile) e osservabile.

La verifica di queste proprietà si può eseguire studiando, rispettivamente, la matrice di raggiungibilità, costruita a partire dalla coppia di matrici (A,B), e la matrice di osservabilità, costruita, invece, a partire dalla coppia di matrici (A,C), del sistema.

Se infatti tali matrici risultano essere di rango pieno, il sistema sarà completamente raggiungibile e completamente osservabile.

Tale verifica si effettua all'inizio di ogni procedura di sintesi affrontata, sia nel tempo discreto che nel tempo continuo, ed è implementata attraverso le seguenti stringhe di codice:

```
n = size(A,1);
Hr = [];
O = [];

% % % % % % % verifica raggiungibilità e osservabilità % % % % %
for i=0:n-1
    Hi= (A^i)*B;
    Hr=[Hr Hi];
    Oi = C*(A^i);
    O = [O; Oi];
end

rango_r=rank(Hr);
rango_o=rank(O);

if rango_r == n
    disp('Il sistema è completamente raggiungibile')
else
    disp('Il sistema non è completamente raggiungibile')
end

if rango_o == n
    disp('Il sistema è completamente osservabile')
else
    disp('Il sistema non è completamente osservabile')
end
```

Nel nostro caso, il sistema risulta essere completamente raggiungibile e completamente osservabile.

3. Sintesi del controllo

L'obiettivo del controllo è quello di stabilizzare il sistema nella sua condizione di equilibrio instabile. Quindi anche a fronte di uno scostamento rispetto alla condizione di riferimento deve poter generare un'azione di controllo che riporti il sistema nella sua condizione di equilibrio.

Per la sintesi del controllo si è considerato uno scostamento di 10 gradi nell'inclinazione del pendolo e per definire i parametri del modello matematico del sistema si è fatto riferimento ad un hoverboard in commercio e al fatto che esso possa essere usato sia da adulti che bambini.

Il controllo è stato sviluppato tramite due tecniche, mettendone a confronto i vantaggi e gli svantaggi.

3.1 Stabilizzazione alla Lyapunov

L'obiettivo di questa fase di progettazione è la realizzazione di una retroazione stabilizzante $u = Kx$ per il sistema, mediante l'equazione di Lyapunov riscritta in termini di disuguaglianze lineari matriciali e comprensiva della retroazione.

L'equazione di Lyapunov si ottiene a partire dalla funzione di Lyapunov nella forma quadratica

$$V(x) = x^T P x$$

con P>0, derivando la quale si arriva, dopo una serie di passaggi, alla sua formulazione in termini di lmi, ovvero:

$$\begin{cases} P > 0 \\ A^T P + PA < 0 \end{cases}$$

Quindi, si vuole trovare una matrice P definita positiva che garantisca che la matrice $A^T P + PA$ sia definita negativa.

Poiché siamo interessati a garantire che il sistema sia asintoticamente stabile a ciclo chiuso, bisogna modificare questo sistema di lmi in modo che tenga conto della retroazione; perciò sfruttando la definita negatività di questa matrice e le proprietà di simmetria della matrice P, si arriva alla formulazione del problema lmi da risolvere, ovvero

$$\begin{cases} QA^T + Y^T B^T + AQ + BY < 0 \\ Q > 0 \end{cases}$$

in cui si sono definite le variabili $Y = KQ$ e $Q = P^{-1}$.

E' un problema di feasibility che una volta risolto (o se risolvibile) restituirà la matrice Q definita positiva e la variabile Y dalla quale ci possiamo ricavare la K che stabilizza il sistema come

$$K = yQ^{-1}.$$

Una volta ottenuto il linearizzato del sistema attorno al punto di equilibrio in esame, (attraverso il comando linmode), si definisce il problema lmi da risolvere con le relative variabili, tramite le seguenti stringhe di codice:

```
setlmis([])

n = size(A,1);
p = size(B,1);

%%%%%%% def variabili %%%%%
Q = lmivar(1,[n 1]);
y = lmivar(2,[1 p]);

%%%%%%%def vincoli %%%%%%
lmiterm([1 1 1 Q],A, 1,'s');  %QA'+AQ<0
lmiterm([1 1 1 y],B,1,'s');   %y'B'+By<0

lmiterm([-2 1 1 Q],1,1);     %Q>0

lmisys=getlmis;
[TMIN,xfeas] = feasp(lmisys)

Qfin = dec2mat(lmisys,xfeas,Q);
yfin = dec2mat(lmisys,xfeas,y);

K = yfin*(Qfin^-1);
eig(A+B*K)

pos=crit_Sylvester(Qfin)
%eig(Qfin)
```

Consiste in un problema di esistenza della soluzione, perciò si richiama il comando **feasp** che avvia una procedura iterativa per risolvere il problema e al termine della quale dirà se il problema è feasible o no.

Nel nostro caso, si ottiene che il problema risulta feasible, ed infatti il valore di *TMIN*, restituito dal comando insieme al vettore delle variabili decisionali *Xfeas*, risulta negativo.

Da *Xfeas*, si estraggono, tramite il comando **dec2mat**, le variabili Q e Y, per poi calcolare la matrice di retroazione K, che nel nostro caso risulta.

Dalla verifica degli autovalori del sistema a ciclo chiuso (A+BK), risulta che la K ottenuta rende il sistema asintoticamente stabile poiché ha autovalori tutti a parte reale negativa.

L'importanza dell'utilizzo delle lmi per la sintesi di un regolatore sta nel fatto che permette di gestire la presenza di eventuali incertezze sul modello del sistema in esame, dovute ad esempio alla variabilità nel valore di alcuni parametri.

Infatti, variando uno o più parametri del modello, si ottengono più realizzazioni del sistema; dalla combinazione lineare dei vari modelli ottenuti si individua un insieme convesso, detto convex hull, di cui ogni modello ne costituisce un vertice.

Sfruttando la formulazione dei vincoli in termini di lmi, è possibile progettare un'unica retroazione che stabilizzi tutti i vari modelli del sistema.

In tal caso, si ottiene la quadratica stabilità, cioè una stabilità che è valida per ciascuna delle configurazioni definite all'interno del convex hull ottenuto dalla combinazione lineare dai vari modelli considerati.

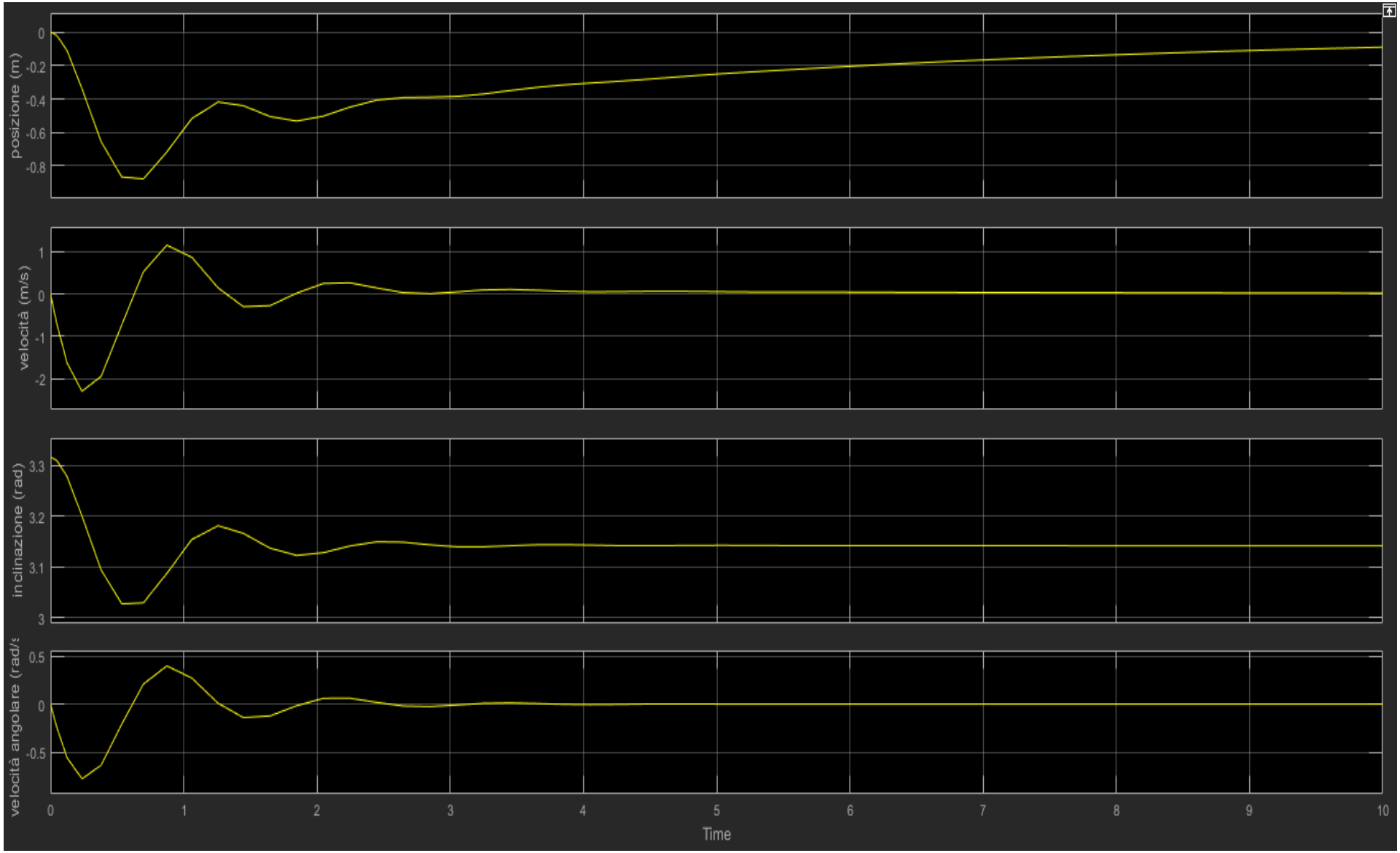
Il problema lmi da risolvere per progettare la retroazione del sistema multimodello è definito, semplicemente replicando i vincoli per ciascuna realizzazione.

Quindi si genera iterativamente, mediante cicli for innestati, un insieme di linearizzati al variare dei parametri scelti e si ottengono così le quadruple di matrici ad essi relative. Al variare di queste matrici si aggiungono dei vincoli al problema lmi e se esiste la soluzione al problema, si otterrà la K che stabilizzerà tutti i modelli e si garantirà la quadratica stabilità.

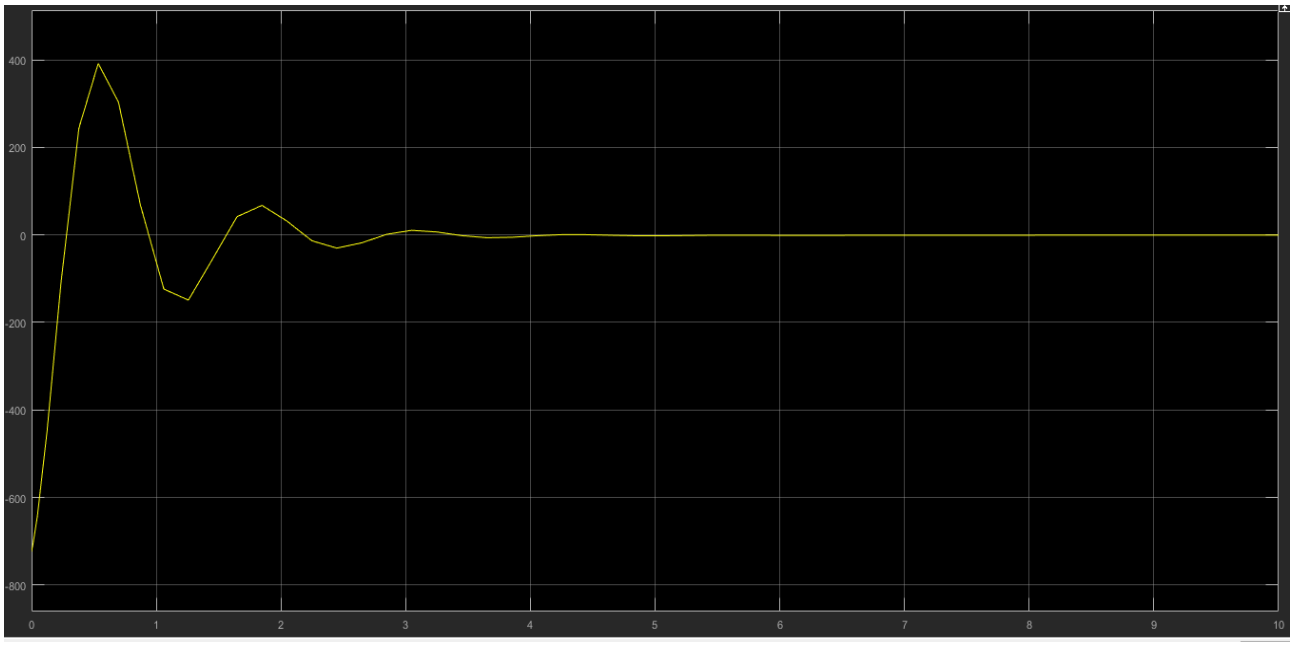
Si è scelto di far variare tre parametri, ovvero il coefficiente di attrito, b, la massa, m, e l'altezza, l, della persona che potrebbe utilizzare il nostro dispositivo:

```
for b=0.2:0.1:0.9
  for m=40:10:100
      for l=1.3:0.1:1.8
```

Una volta individuata la matrice di retroazione, si è simulato l'azione di controllo da essa generata per una tra le possibili rappresentazioni del sistema definite. In particolare la simulazione è stata effettuata sul sistema non lineare, ottenendo il seguente andamento nelle variabili di stato:


posizione (m)
0
-0.2
-0.4
-0.6
-0.8
velocità (m/s)
1
0
-1
-2
inclinazione (rad)
3.3
3.2
3.1
3
velocità angolare (rad/s
0.5
0
-0.5
0
1
2
3
4
5
6
7
8
9
10
Time

Mentre l’andamento dell’azione di controllo è il seguente:



Infatti, dalla simulazione del controllo si desume che pur avendo alcune dinamiche degli stati abbastanza veloci, si ottengono pure degli andamenti che presentano delle sovraelongazioni e un elevato valore della variabile di controllo (circa 700 N) sulla quale non si può agire in alcun modo tramite questa tecnica. Né si potrebbe provare a modificare i tempi di evoluzione di alcuni variabili come, ad esempio, la posizione.

Infatti un limite della sintesi tramite Lyapunov è che non si possono definire delle prestazioni; perciò si è preferito utilizzare un’altra tecnica per la sintesi del controllo: ovvero il controllo LQ.

3.2 Controllo LQ

Il controllo LQ è una tecnica di controllo che, in confronto alla stabilizzazione tramite Lyapunov, oltre a garantire la stabilità del sistema in esame consente anche di definire delle prestazioni.

Esso permette di minimizzare un indice di prestazione, J, attraverso cui sarà possibile assegnare un determinato comportamento al processo che si vuole regolare e ottimizzare il controllo.

Si dovrà quindi risolvere un problema di minimizzazione sfruttando nello specifico delle matrici di peso da dimensionare in modo opportuno fino a quando non si otterranno le prestazioni desiderate.

Le matrici su cui si può agire sono:

- La matrice S, simmetrica e semidefinita positiva, tramite la quale si pesa lo scostamento dello stato finale del sistema dallo zero
- La matrice R_x, anch'essa simmetrica e semidefinita positiva, tramite la quale si pesa lo scostamento dello stato dallo zero durante il transitorio 0 – T (con T finito o infinito)
- La matrice R_u, simmetrica e definita positiva, con la quale si pesa lo sforzo di controllo e si agisce sulla moderazione della variabile di controllo.

L'obiettivo sarà progettare un regolatore che produca un'azione di controllo che rende minimo l'indice di prestazione, e consente di regolare lo stato a zero.

Questa tecnica di controllo può essere affrontata, col medesimo approccio, sia su orizzonte temporale finito sia su orizzonte infinito particolarizzando questi casi sia per sistemi tempo discreto che sistemi tempo continuo.

Poiché interessati ad ottenere una strategia di controllo stabilizzante, per la sintesi del regolatore LQ si è considerata solo la trattazione su orizzonte infinito.

L'indice di costo da minimizzare, J, è di tipo quadratico ed è funzione della condizione iniziale e del forzamento (o successione di forzamenti) applicato nell'intervallo temporale [0,T) (con T finito o infinito). Nel caso di orizzonte infinito, è così definito:

- Nel tempo discreto: $J(x_o, u(.)_{[0,\infty)}) = \sum_{k=0}^{\infty} x_k^T R_x x_k + u_k^T R_u u_k + x_\infty^T S x_\infty$
- Nel tempo continuo: $J(x_o, u(.)_{[0,\infty)}) = \int_0^\infty x(t)^T R_x x(t) + u(t)^T R_u u(t)\, dt$

In particolare, notiamo l'assenza della matrice di peso S nell'espressione di costo; infatti in questo caso, il suo valore non è importante perché lo stato terminale è x_∞ il quale deve essere nullo in modo da garantire che l'indice di costo sia limitato. Ciò porta anche alla sintesi di una strategia di controllo che risulterà stabilizzante perché qualsiasi sia la condizione iniziale si arriverà comunque alla condizione di $x_\infty = 0$.

Dopo una serie di manipolazioni e semplificazioni sulla relazione che definisce l'indice di costo, si conclude che il valore minimo che esso può assumere coincide con

$$J = x_0^T M x_0$$

e che la legge di controllo ottimo si può esprimere in termini di retroazione dello stato, ovvero

$$u = -Kx$$

Si ottiene così una retroazione statica dello stato.

L'espressione che definisce J è funzione delle matrici di peso e di una matrice, definita positiva, M.

La matrice M è una matrice fissa che si ottiene risolvendo l'equazione algebrica di Riccati:

L'equazione algebrica di Riccati da risolvere è

- Nel tempo discreto: $M = R_x + F^T MF - F^T MG(R_u + G^T MG)^{-1} G^T MF$
- Nel tempo continuo: $MF + F^T M + R_x - MGR_u{}^{-1} G^T M = 0$

Trovata la M, si può definire il minimo valore dell'indice di costo e la matrice di retroazione K:

- Nel tempo discreto: $K = (R_u + G^T MG)^{-1} G^T MF$
- Nel tempo continuo: $K = R_u{}^{-1} G^T M$

Infine, è possibile riformulare il problema di controllo LQ anche in termini di disuguaglianze lineari matriciali in modo da poter realizzare la sintesi di un regolatore anche nel caso di sistemi aventi incertezza di tipo politopica, cioè nel caso in cui si ha una rappresentazione multimodello del sistema in esame, cosa che non sarebbe stata possibile fare con la risoluzione dell'equazione di Riccati.

Si è scelto di affrontare la sintesi del controllo in termini di disuguaglianze lineari matriciali, in modo da poter regolare il nostro sistema in più casi possibili e in modo più realistico, proprio perché un hoverboard può essere utilizzato sia da adulti che da bambini.

In particolare, si è considerato solo il caso tempo discreto in cui la definizione del problema lmi parte dal supporre verificata la condizione:

$$\Delta V \leq -(x_k^T R_x x_k + x_k{}^T R_u u_k)$$

Con ΔV variazione al passo della funzione di Lyapunov $V = x_k^T P x_k$ con P>0

La funzione obiettivo in questo caso è la V(0), minimizzando la quale si otterrà la minimizzazione dell'indice di costo.

Definendo la $V(0) = x_0^T P x_o$ che indichiamo con γ, il problema di minimizzazione da risolvere è:

$$\begin{cases} \min \gamma \\ s.t. \\ x_0^T P x_0 \leq \gamma \\ P > 0 \\ \Delta V \leq -(x_k^T R_x x_k + {x_k}^T R_u u_k) \end{cases}$$

Questo è un problema agli autovalori ma va riscritto in termini di disuguaglianze lineari matriciali, bisogna perciò modificare il primo e il terzo vincolo.

Considerando un generico sistema tempo discreto $x(k+1) = Fx(k) + Gu(k)$, un'azione di controllo nella notazione $u = -Kx$ e manipolando il terzo vincolo si ottiene che la sua rappresentazione in termini lmi è:

$$\begin{bmatrix} Q & QF^T + y^T G^T & QR_x^{\frac{1}{2}} & y^T R_u^{\frac{1}{2}} \\ * & Q & 0 & 0 \\ * & * & I & 0 \\ * & * & 0 & I \end{bmatrix} \geq 0 \qquad (*)$$

in cui si sono definite le variabili $y = KQ$ e $Q = P^{-1}$.

Il primo vincolo, applicando il **complemento di Schur** viene riscritto come

$$\begin{pmatrix} \gamma & x_0^T \\ * & Q \end{pmatrix} \geq 0$$

$$Q > 0$$

Il complemento di Schur permette di definire una disuguaglianza matriciale non lineare nella forma $Q - S^T R^{-1} S > 0$ con R>0, in una disuguaglianza lineare nella forma

$$\begin{pmatrix} Q & S^T \\ * & R \end{pmatrix} \geq 0$$

Dunque il problema di ottimizzazione, alla fine consisterà nel minimizzare γ, al variare di Q, γ e y, soggetto ai vincoli lmi definiti sopra.

Questo problema di minimizzazione si risolve in matlab tramite il comando **mincx** che trova il minimo valore della funzione obiettivo soggetta ai vincoli lmi definiti.

```
n = decnbr(lmisys);
c = zeros(n,1);

for j=1:n
    gamma_j=defcx(lmisys,j,gamma);
    c(j)=gamma_j;
end

[COPT,XOPT] = mincx(lmisys,c,[0 300 0 0 0]);
gamma = dec2mat(lmisys,XOPT,gamma);

Qfin = dec2mat(lmisys,XOPT,Q);
yfin = dec2mat(lmisys,XOPT,y);

K = yfin*(Qfin^-1);

mod=abs(eig(Ad+Bd*K));
```

Risolto questo problema di minimizzazione avremo come soluzione le variabili Q e y, tramite le quali potremo ricavarci la retroazione ottima K definita come $K = yQ^{-1}$, la quale stabilizzerà il sistema.

Tramite la formulazione in disuguaglianze lineari matriciali è possibile estendere il problema al caso di sistemi aventi incertezza politopica poiché basterà definire un nel problema lmi tanti vicoli relativi a ΔV, cioè i vincoli (*), per ciascuno dei vertici del politopo. Infatti, l'unica quantità che varia in funzione dei vari modelli è ΔV, mentre Q e γ, non sono funzione del sistema e i vincoli relativi ad essi rimangono tali.

Per la realizzazione del sistema multimodello sono stati variati gli stessi parametri considerati nella tecnica di stabilizzazione tramite Lyapunov.

Il regolatore LQ si progetta tarando in modo opportuno le matrici di peso R_x e R_u, i cui valori dipendono dalle prestazioni che si vogliono ottenere

A livello pratico, la matrice R_x è una matrice diagonale che pesa lo scostamento dello stato; aumentandone il valore infatti si rende meno accettabile lo scostamento delle variabile di stato durante il transitorio, ottenendo dinamiche più veloci (tempi di evoluzione minori)

La matrice R_u, invece permette di agire sulla variabile di controllo e moderarne il valore e in questo caso è una matrice 1x1 poiché l'azione di controllo coincide con la forza applicata al sistema dal motore.

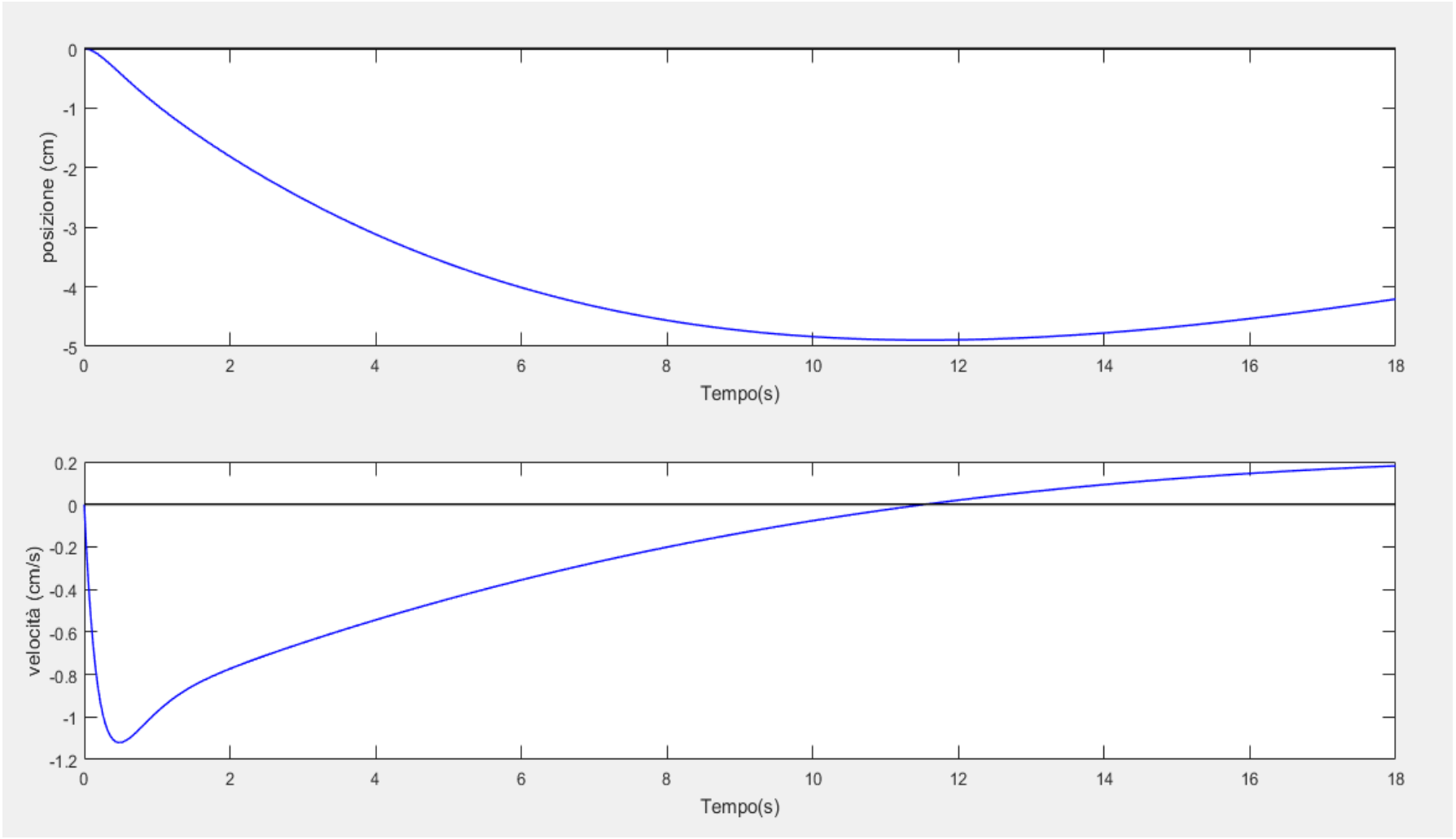
La taratura di queste matrici viene fatta mediante simulazioni, cambiandone di volta in volta i valori in base ai risultati ottenuti.

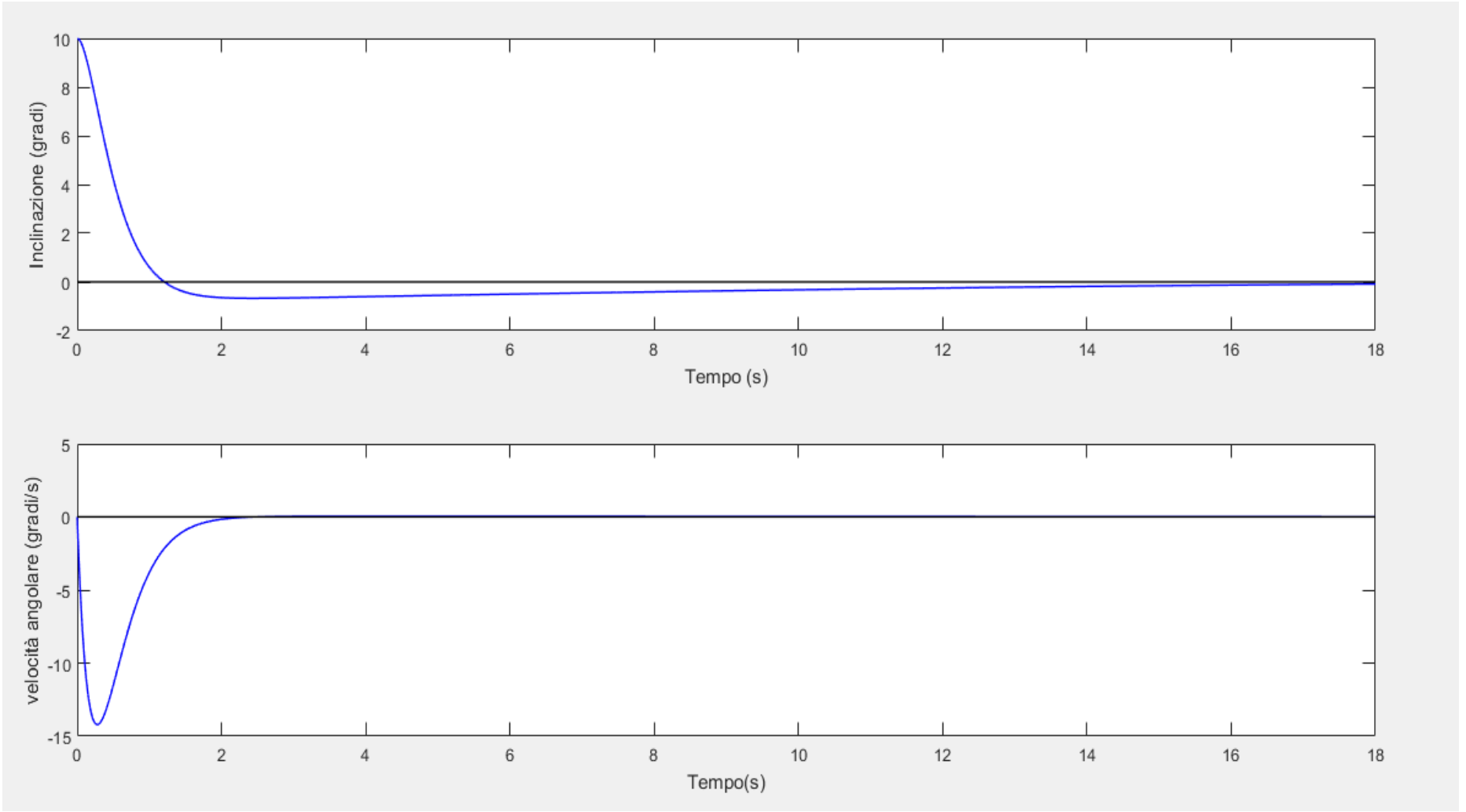
Le simulazioni sono realizzate su matlab sempre considerando uno scostamento iniziale di 10 gradi rispetto alla condizione di equilibrio.

Come prima cosa si deve discretizzare, con un tempo di campionamento di 1 ms il sistema linearizzato tramite il comando **c2d** che appunto converte un sistema tempo continuo in un sistema tempo discreto.

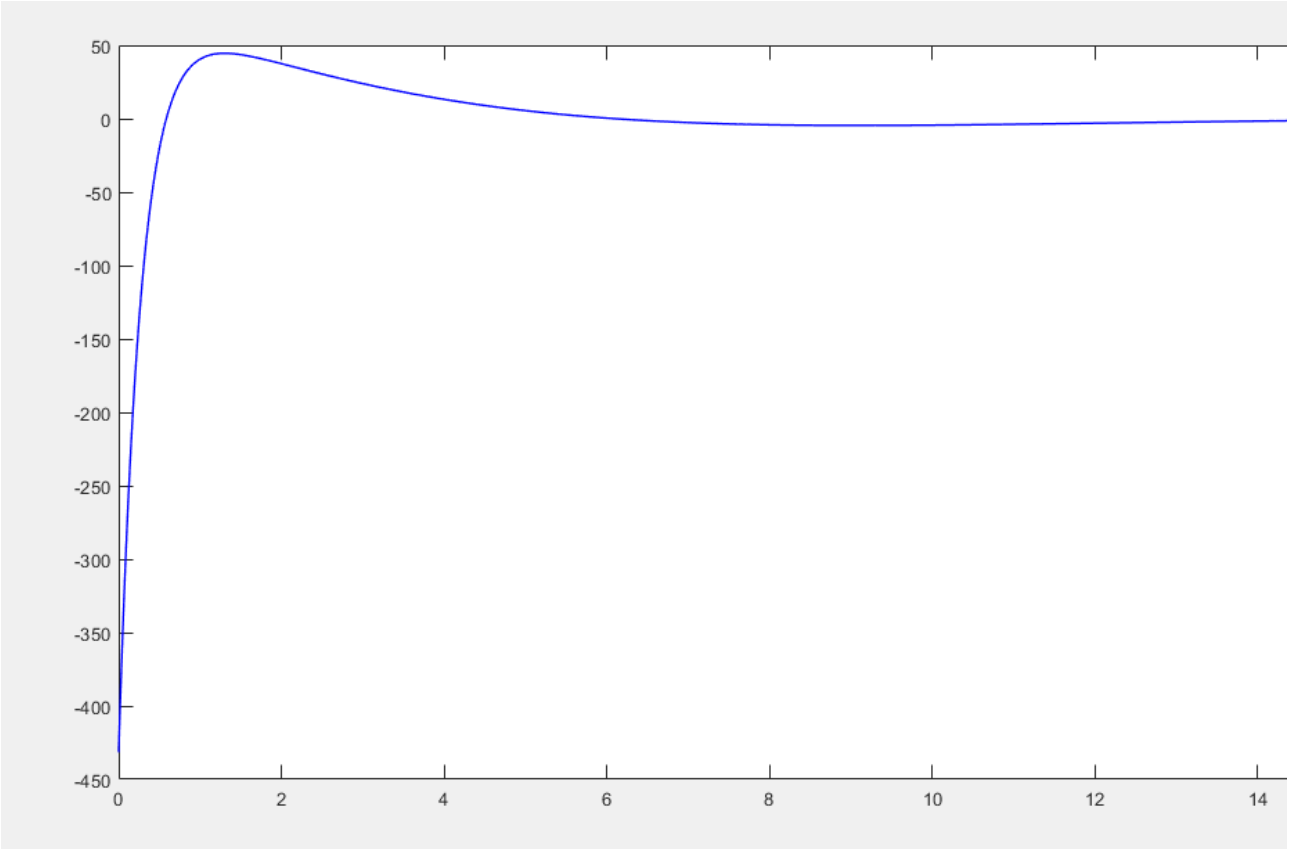
La prima simulazione permette di valutare gli andamenti delle variabili di stato ponendo R_u=1 e R_x pari ad una matrice identità 4x4,poichè 4 sono gli stati del sistema, e il problema lmi viene definito all'interno di cicli for tramite le seguenti righe di codice:

```
    % % % matrici di peso %%%%
    alpha_x = 1;
    alpha_u = 1;
    %V=[900 100 700 1];
    %Rx =diag(V);
    Ru = alpha_u*eye(size(Bd,2));
    n = size(Ad,1); p = size(Bd,1);
    dtheta = 10; x0_inf = [0;0;(dtheta/57.3);0];

    setlmis([])
    %%%%%%% def variabili %%%%%
    Q = lmivar(1,[n 1]);
    gamma = lmivar(1,[1 1]);     %verificare le dimensioni
    y = lmivar(2,[1 p]);
        %%%%%%%def vincoli %%%%%%
    lmiterm([-1 1 1 gamma],1,1);          %gamma
    lmiterm([-1 1 2 0],x0_inf');          %x0_inf'
    lmiterm([-1 2 2 Q],1, 1);             %Q

    lmiterm([-2 1 1 Q],1,1);              %Q
    lmiterm([-2 1 2 Q],1,Ad');            %QAd'
    lmiterm([-2 1 2 -y],1,Bd');           %y'Bd
    lmiterm([-2 1 3 Q],1,Rx^(1/2));       %QRx^(1/2)
    lmiterm([-2 1 4 -y],1,Ru^(1/2));      %y'Ru^(1/2)
    lmiterm([-2 2 2 Q],1,1);              %Q
    lmiterm([-2 3 3 0],1);                %I
    lmiterm([-2 4 4 0],1);                %I
  end
end
end
```


posizione (cm)
0
-1
-2
-3
-4
-5
0
2
4
6
8
10
12
14
16
18
Tempo(s)
velocità (cm/s)
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
0
2
4
6
8
10
12
14
16
18
Tempo(s)


Inclinazione (gradi)
10
8
6
4
2
0
-2
0
2
4
6
8
10
12
14
16
18
Tempo (s)
velocità angolare (gradi/s)
5
0
-5
-10
-15
0
2
4
6
8
10
12
14
16
18
Tempo(s)

Mentre la variabile di controllo risulta



Come si può vedere dai grafici, il problema di questo controllo consiste non tanto nel valore dell'azione di controllo quanto nelle dinamiche delle variabili di stato, in quanto alcune di esse presentano tempi di assestamento molto lunghi.

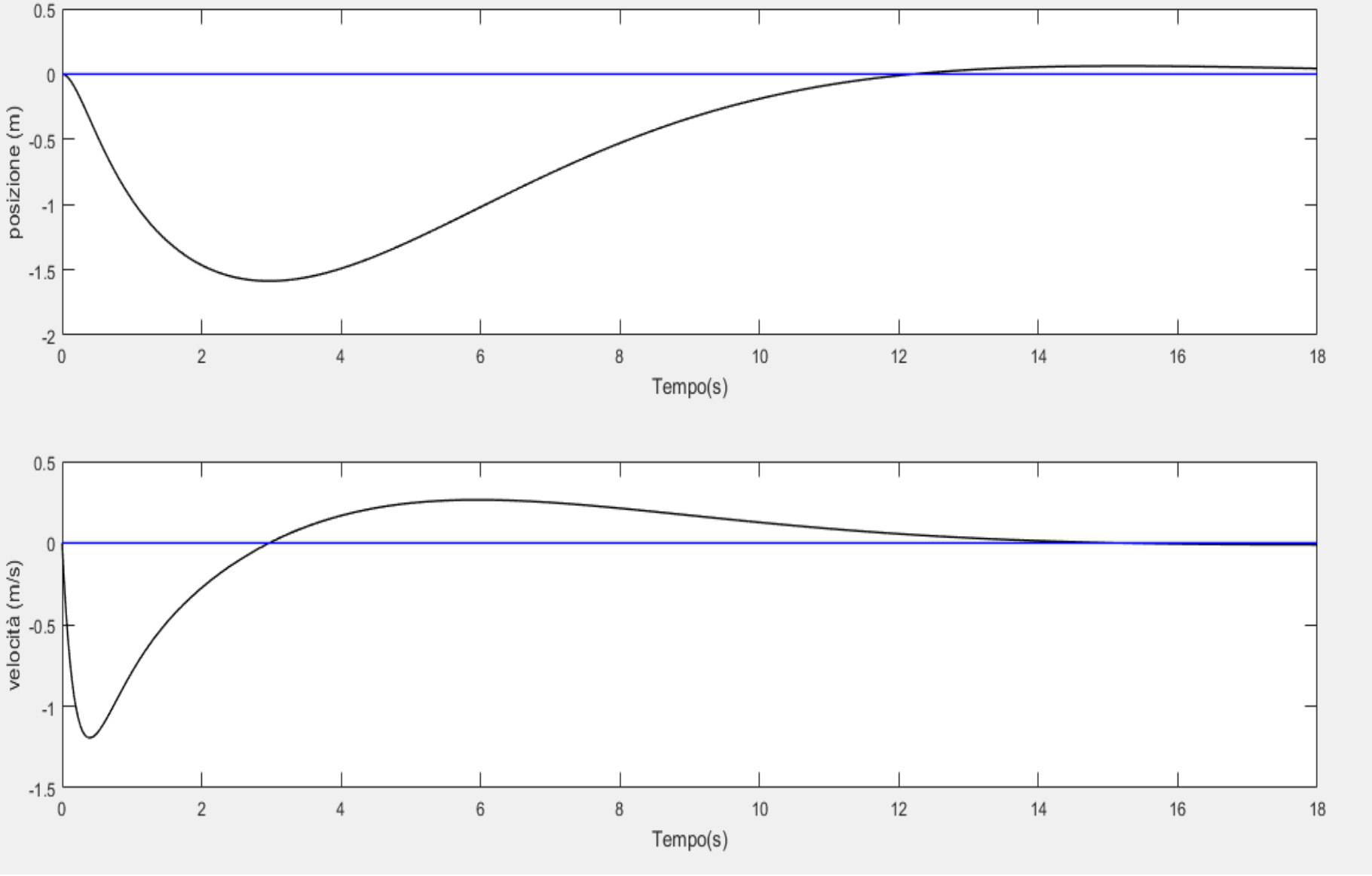
Perciò, per ottenere tempi di evoluzione maggiori, si modifica principalmente la matrice R_x che viene espressa come una matrice diagonale con elementi di valore diverso in modo da dare maggior peso alle dinamiche più lente del sistema.

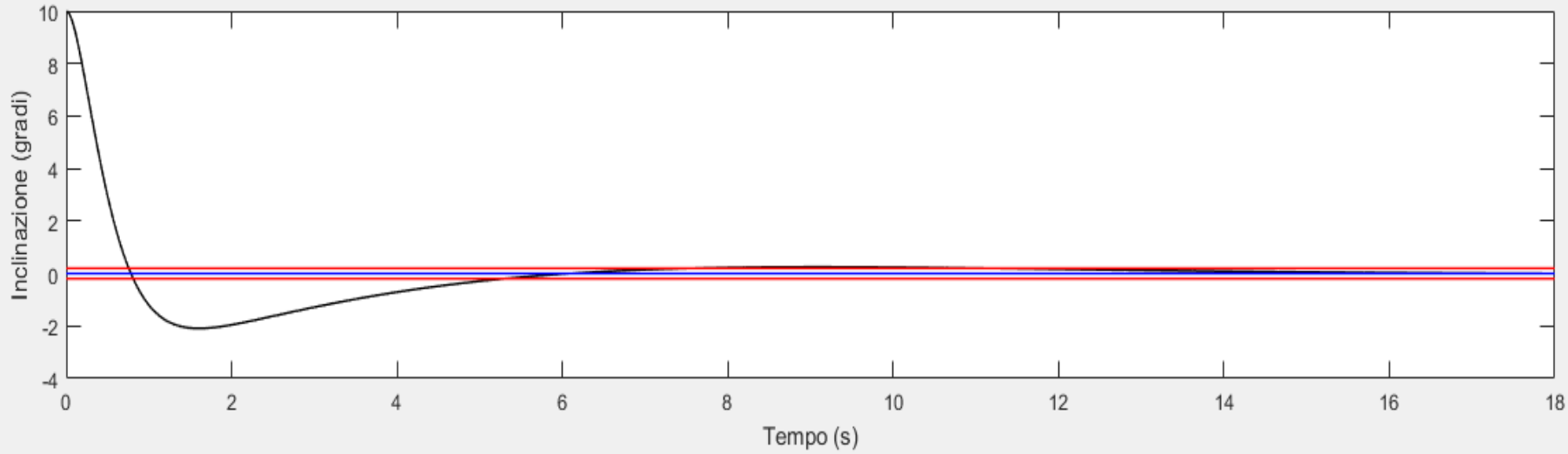
Dalle varie simulazioni effettuate si conclude che gli effetti dei valori associati alle due matrici sono contrastanti tra loro poiché aumentando di molto R_x si ottengono tempi di assestamento più veloci ma oscillazioni nell'andamento degli stati e un aumento del modulo dell'azione di controllo. Invece, aumentando di molto R_u si riesce ad contenere il valore dell'azione di controllo rallentando però le dinamiche degli stati.

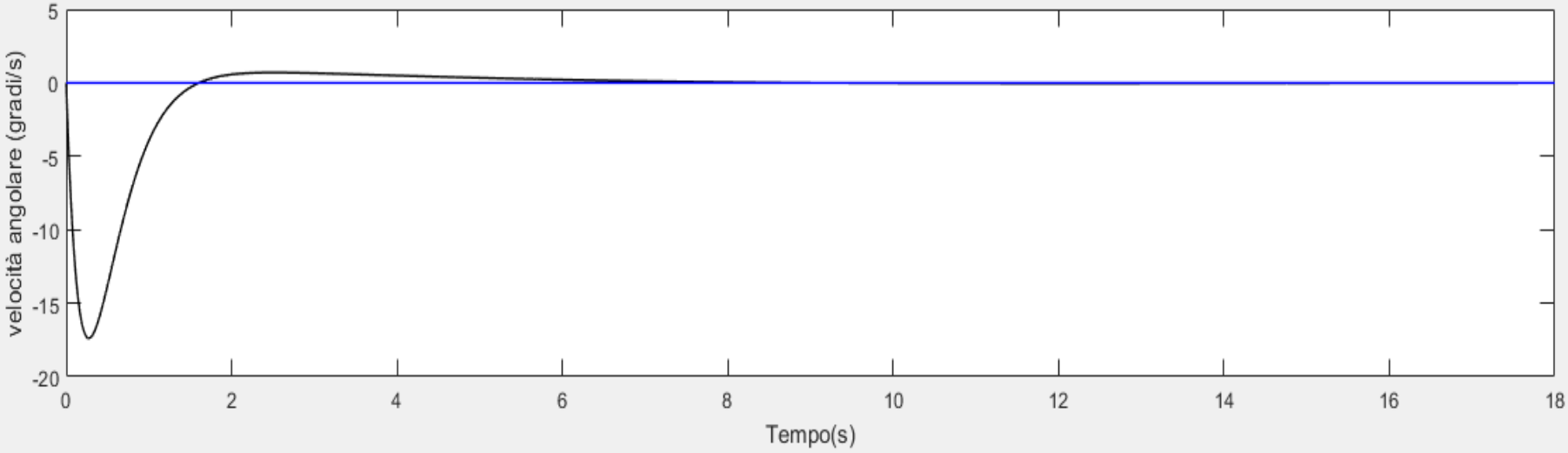
Alla fine si è ottenuto un comportamento accettabile per la seguente combinazione di matrici :

$$R_x = diag(900,100,600,1) \text{ e } R_u = 4$$

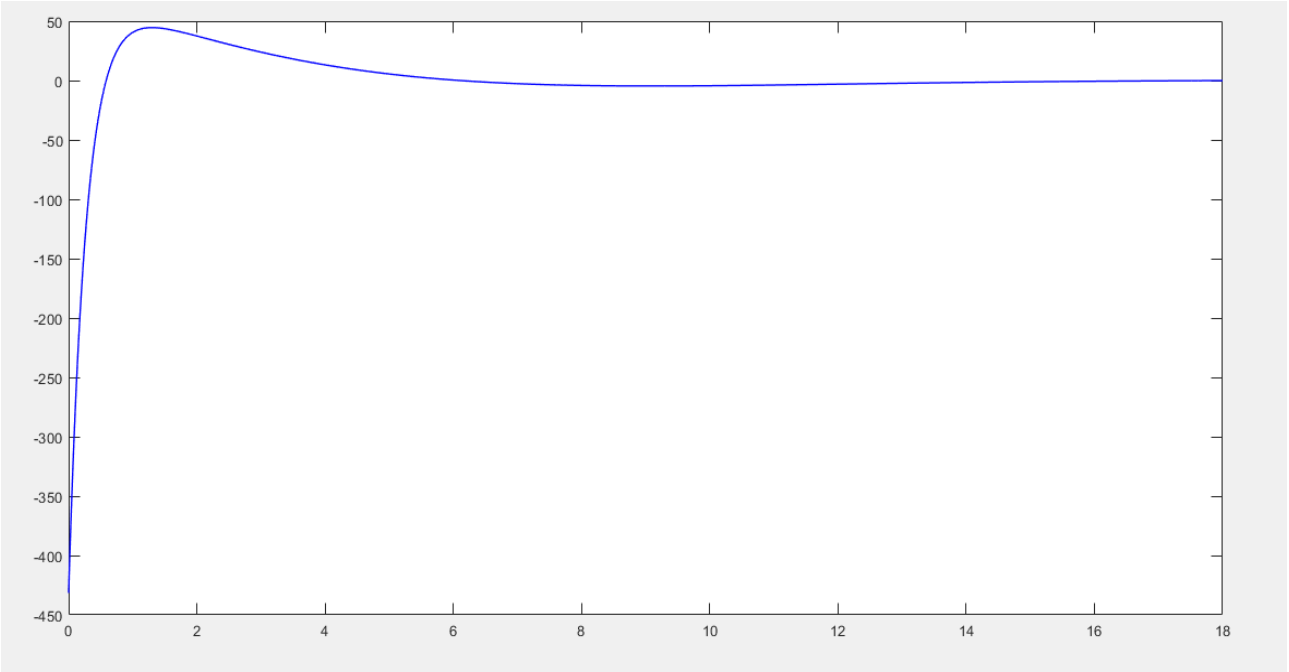
Ottenendo tempi di assestamento minori per le variabili di stato, in particolare un tempo di assestamento al 20% pari a 5 secondi per l'inclinazione del pendolo.


0.5
0
-0.5
-1
-1.5
-2
posizione (m)
0
2
4
6
8
10
12
14
16
18
Tempo(s)
0.5
0
-0.5
-1
-1.5
velocità (m/s)
0
2
4
6
8
10
12
14
16
18
Tempo(s)


Inclinazione (gradi)
10
8
6
4
2
0
-2
-4
0
2
4
6
8
10
12
14
16
18
Tempo (s)


velocità angolare (gradi/s)
5
0
-5
-10
-15
-20
0
2
4
6
8
10
12
14
16
18
Tempo(s)

Per quanto riguarda l'azione di controllo invece si ha il seguente andamento:



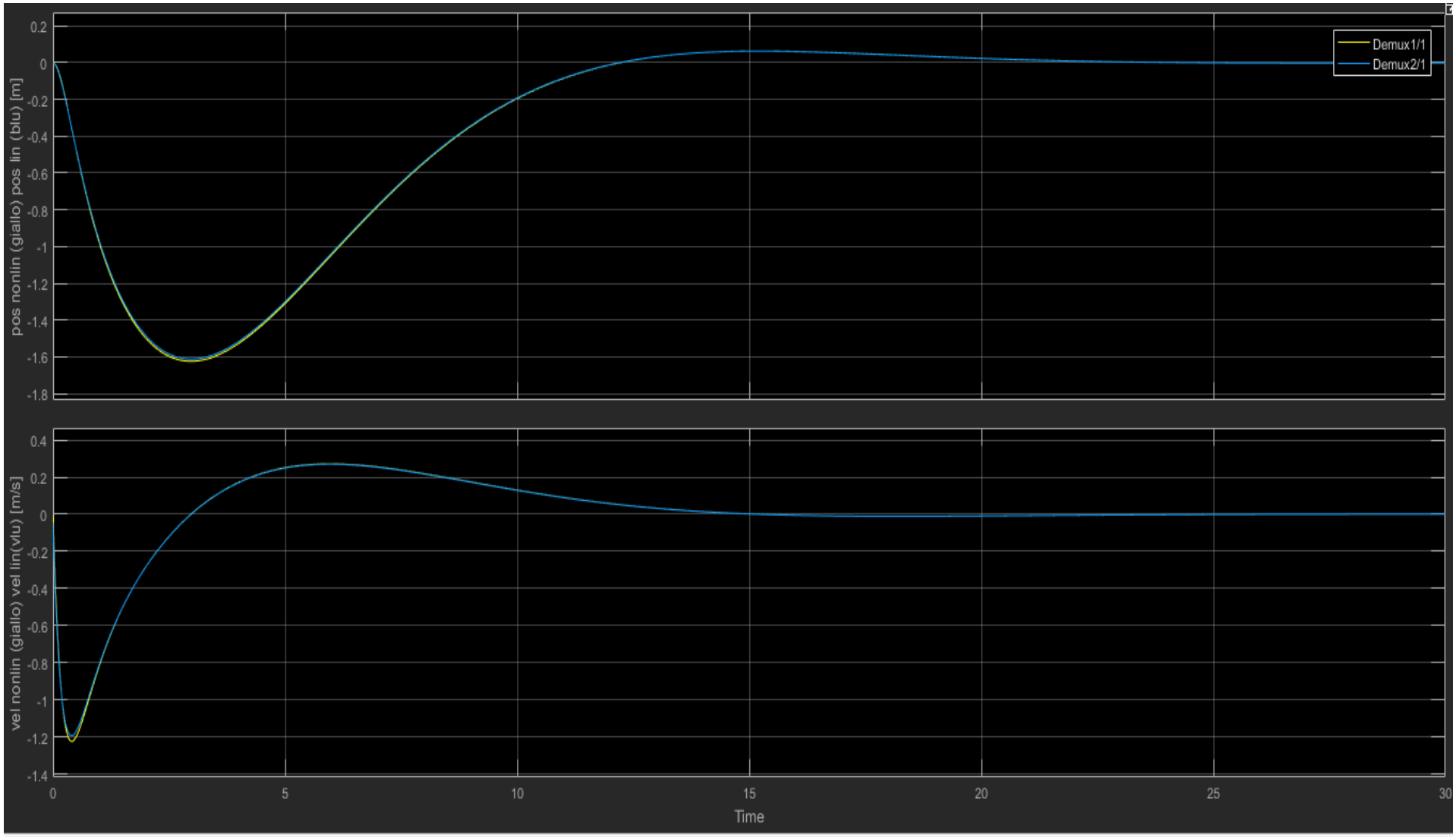
Presenta un valore massimo pari a circa 430 N che comunque risulta accettabile visto che corrisponde ad una coppia di circa 70 watt per dispositivi aventi ruote da 6.5''(43 watt per ruote da 10'') e che comunque è sostenibile dai motori elettrici e anche da alcuni modelli di Hoverboard in commercio.

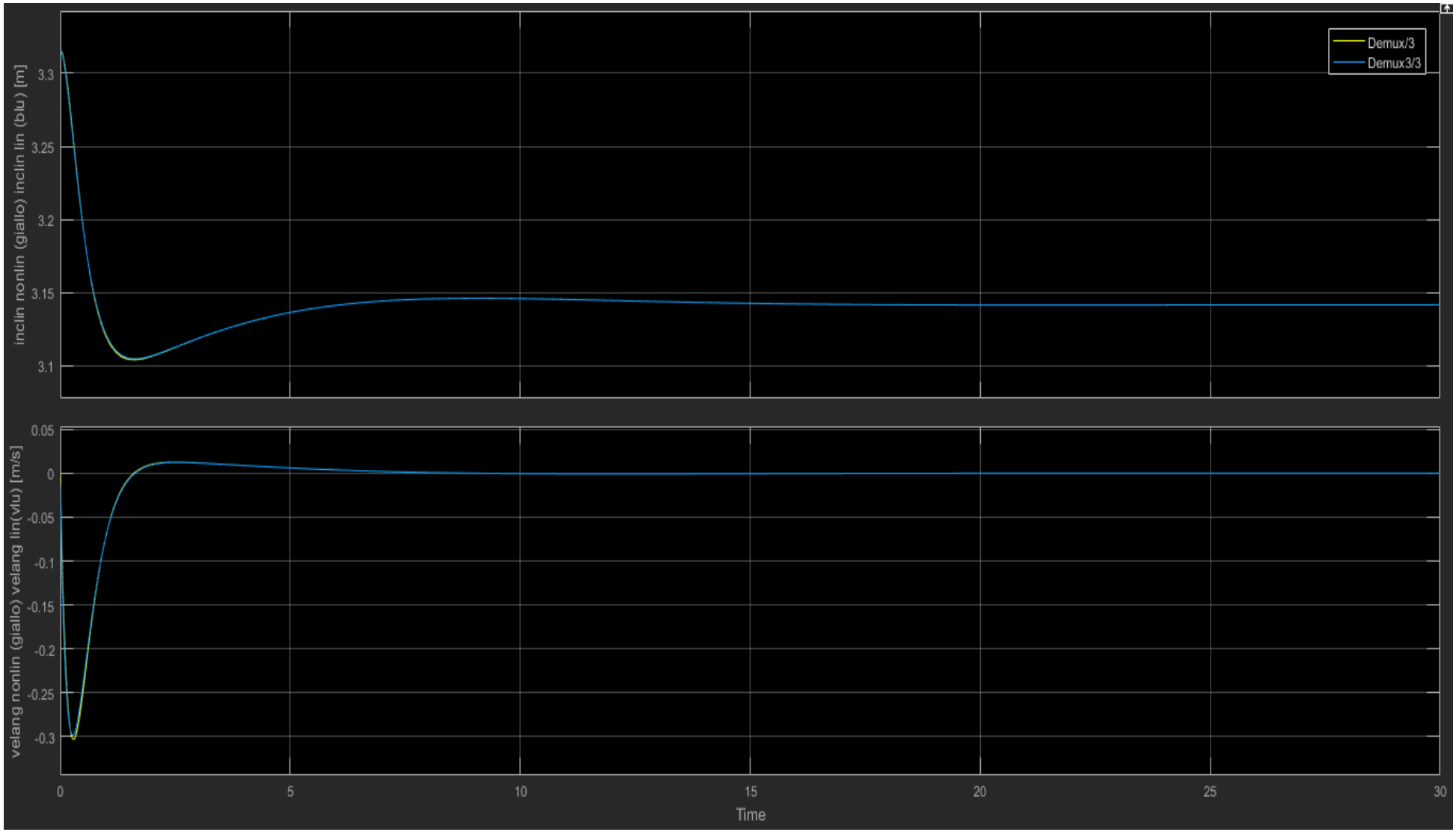
La matrice di retroazione così ottenuta è

```
K =

   1.0e+03 *

    0.0144    0.0653   -2.4731   -0.7899
```

e stabilizza, di fatto, il sistema poiché gli autovalori a ciclo chiuso sono tutti in modulo minore di 1.

Si è verificato poi la validità del controllo applicandolo anche al sistema non lineare attraverso una simulazione su simulink. Mettendo a confronto l'effetto del controllo sul sistema non lineare e sul sitema linearizzato si ottengono i seguenti andamenti:


Demux1/1
Demux2/1
pos nonlin (giallo) pos lin (blu) [m]
vel nonlin (giallo) vel lin(vlu) [m/s]
Time
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
-1.8
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
0
5
10
15
20
25
30


Demux/3
Demux3/3
inclin nonlin (giallo) inclin lin (blu) [m]
3.3
3.25
3.2
3.15
3.1
velang nonlin (giallo) velang lin(vlu) [m/s]
0.05
0
-0.05
-0.1
-0.15
-0.2
-0.25
-0.3
0
5
10
15
20
25
30
Time

Gli andamenti dei 2 sistemi risultano molto simili, ciò dimostra l'efficacia del regolatore progettato.

Il minimo valore dell'indice di costo restituito dal comando mincx risulta pari a 1.1093*10^7 e la matrice R_x ottenuta risulta definita positiva e la matrice $R_u$risulta definita come ρI , allora il controllo LQ progettato risulta essere robusto.

4. Controllo vincolato

4.1 Command governor

Nel progetto di un controllo di un sistema molto spesso ci sono dei vincoli (di attuazione,..) di cui tener conto; quindi è importante garantire stabilità e prestazioni ma è importante anche garantire il rispetto di questi vincoli al fine di avere il corretto funzionamento del sistema.

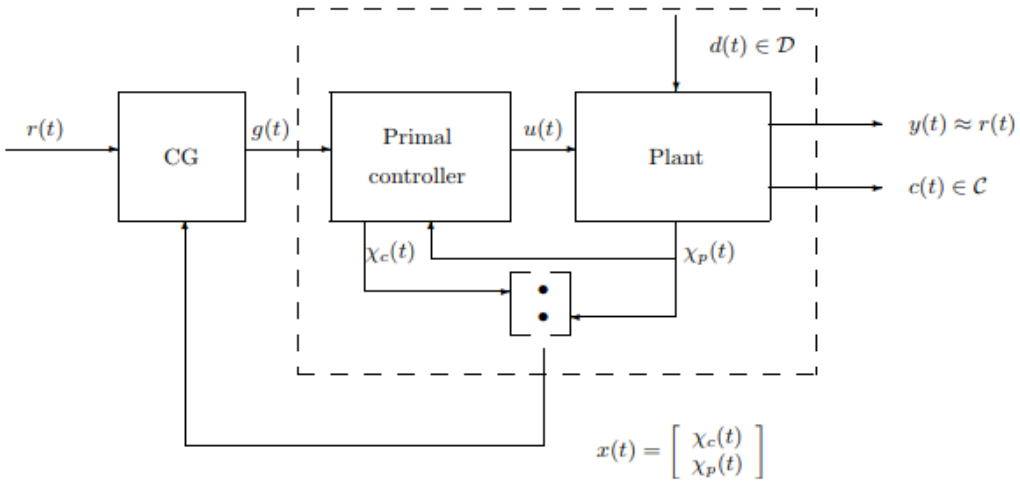
La gestione dei vincoli la effettuiamo tramite una tecnica di controllo predittivo detta Command Governor, o gestore del riferimento.

Il command governor si interpone tra il sistema e il riferimento ad esso imposto dall'utilizzatore e serve a garantire che i vincoli necessari al corretto funzionamento siano rispettati anche a fronte di un eventuale riferimento che potrebbe portare ad una loro violazione.

Se necessario, perciò il command governor modificherà il riferimento in modo da evitare la violazione dei vincoli; mentre nel caso in cui i vincoli siano rispettati fornirà esattamente il riferimento.

Il sistema su cui agisce è comunque il sistema precompensato, cioè comprensivo di un eventuale controllo che ne garantisca l'asintotica stabilità a ciclo chiuso e che definisca le prestazioni del sistema in caso di rispetto dei vincoli.

Lo schema di un generico sistema che include il command governor è



Il command governor quindi in base allo stato del sistema e del riferimento r(t) applicato, definisce tra i vari ingressi ammissibili quello che rappresenta la migliore approssimazione del riferimento, indicato con g(t), compatibile con i vincoli e che ne garantirà il soddisfacimento per ogni istante $k \geq 0$.

Il sistema precompensato deve essere inoltre offset free, così che in assenza di violazione dei vincoli il segnale in uscita dal sistema sia proprio quello applicato.

La g(t) più simile ad r(t) è definita in termini di norma, infatti

$$g(t) = \arg\min_{w} \|w - r(t)\|_{\psi}^{2}$$

soggetta ad un insieme di vincoli e con w appartenente all'insieme degli ingressi ammissibili.

L'insieme di appartenenza dei vincoli è un insieme convesso ed è definito come un'intersezione di semispazi. Inoltre, si dimostra essere finito poiché grazie alla proprietà di asintotica stabilità del sistema precompensato, se i vincoli sono rispettati fino ad un certo istante $\bar{k}$, allora lo saranno per ogni $k \geq 0$.

Il problema da risolvere è un problema di minimizzazione quadratica, risolto in matlab con il comando **quadprog**.

Prima di risolvere il problema, però, bisogna definire $\bar{k}$ e lo si fa tramite un algoritmo iterativo: in pratica, la ricerca si effettua tramite un ciclo for che ad ogni iterazione incrementa una variabile k e verifica una condizione. Il ciclo termina appena la condizione è verificata e $\bar{k}$ sarà proprio pari al valore della variabile k.

La condizione da verificare deriva dalla risoluzione di un problema di programmazione lineare risolto in matlab tramite il comando **linprog**.

Il command governor deve essere sempre attivo e risolvere ad ogni passo di campionamento il problema di minimizzazione: esso infatti ottenuta la g(t) ottima all'istante k, la applica al sistema in tale istante, rimisura lo stato del sistema e risolve di nuovo il problema di minimizzazione poiché lo stato e il riferimento potrebbero essere cambiati, ottenendo la nuova g(t) che più si avvicina al riferimento e rispetta i vincoli.

Il sistema di cui si è progettato il controllo rappresenta un dispositivo, un sistema reale che è soggetto a dei vincoli; esso deve poter essere utilizzato con sicurezza e non deve essere soggetto a un forza eccessiva che comprometta il funzionamento dei dispositivi che lo compongono.

Da qui la necessità di affrontare anche un problema di controllo vincolato e quindi implementare il command governor da applicare al nostro sistema precompensato.

La progettazione del command governor che è affrontata consiste solo nell'implementazione del codice che ne permette il funzionamento e non in una simulazione pratica dello stesso.

Si definiscono perciò gli elementi necessari a definire la strategia del command governor, facendo riferimento alla rappresentazione isu tempo discreta di un sistema precompensato

Nel nostro caso, il controllo è rappresentato da una retroazione statica dello stato di tipo LQ, quindi l'ingresso è solo funzione dello stato; tuttavia si inserisce un ulteriore guadagno, Kg, per rendere il sistema offset free: partendo dalla stessa condizione di offset, esso è definito come quella matrice che ci permette di garantire questa condizione. Questo guadagno lega l'ingresso del sistema con la g(t) fornita dal Command governor.

Bisogna poi definire l'uscita vincolata: la matrice Hc è costituita dalla matrice di retroazione e dall'uscita del sistema che si vuole controllare che nel nostro caso è rappresentata dall'inclinazione del pendolo; mentre la matrice L è costituita dal guadagno Kg e da un certo numero di zeri che renda le sue dimensioni compatibili ai fini dei calcoli.

Una volta verifica l'ipotesi di asintotica stabilità, si definiscono i vincoli: si è deciso di imporre dei vincoli sulla massima inclinazione che si può assegnare al pendolo ($|\theta| \leq 15$ gradi) e sul valore della forza che il sistema può ricevere in ingresso, stabilito in base al massimo valore dell'azione di controllo ottenuto in fase di sintesi del regolatore ($|F| \leq 400\ N$).

Si definisce dunque una matrice T che comprende i segni dei vincoli e un vettore g in cui invece inserirne i valori.

Il tutto è effettuato tramite le seguenti righe di codice:

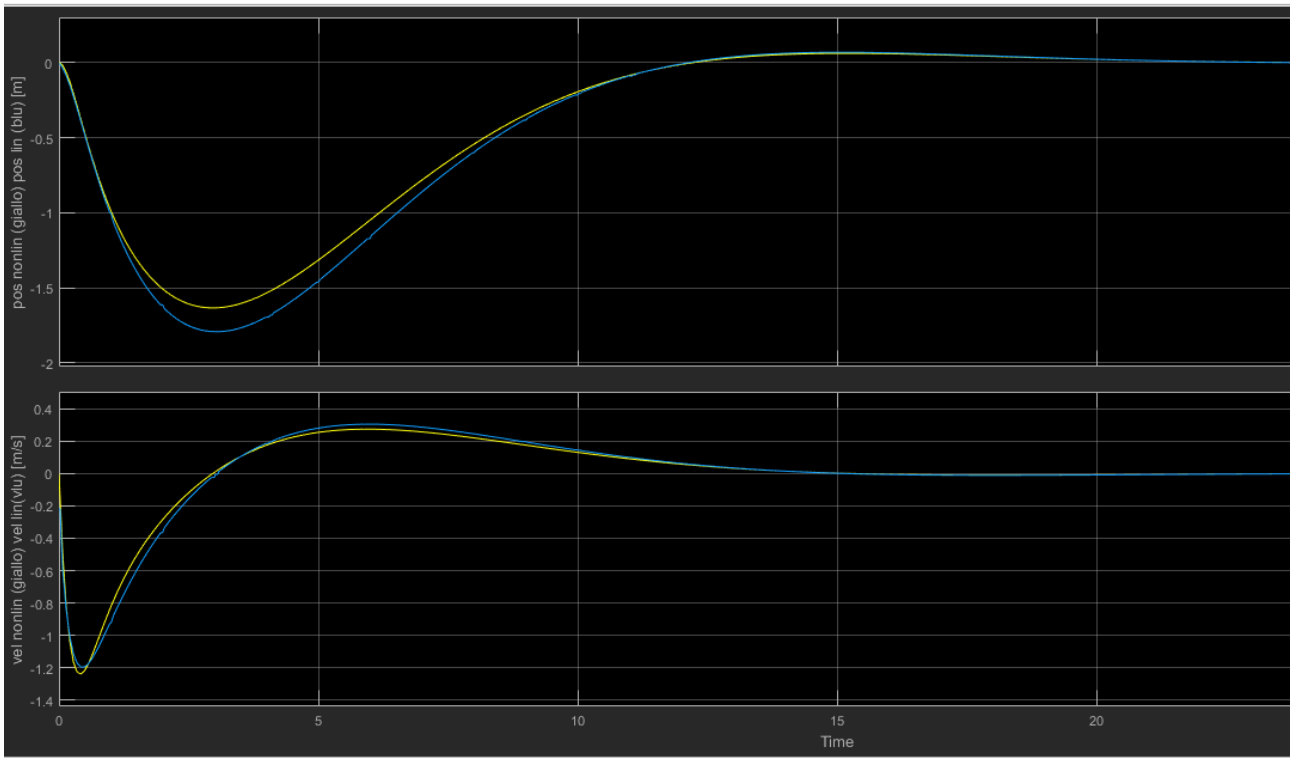
```
Kx = K;

PHI = Ad+Bd*K;
Hy=Cd+Dd*K;

%%%%%%%verifica proprietà di asintotica stabilità
eig(PHI)
%%%%%%%%verifica offset free%%%%%%%%
I = eye(length(PHI));
temp=Hy*inv(I-PHI)*Bd;
Kg=(1/temp);
G=Bd*Kg;
Im = Hy*inv(I-PHI)*G;
norm(Im);

L = [Kg; zeros(size(Kg))];

T_Hb = [0 0 1 0];
Hc = [Kx;T_Hb];

%%%%%vincoli su V e T
T =[1 0 ;-1 0 ;0 1 ;0 -1];
g = [(500) (500) (15/57.3) (15/57.3)]';
```

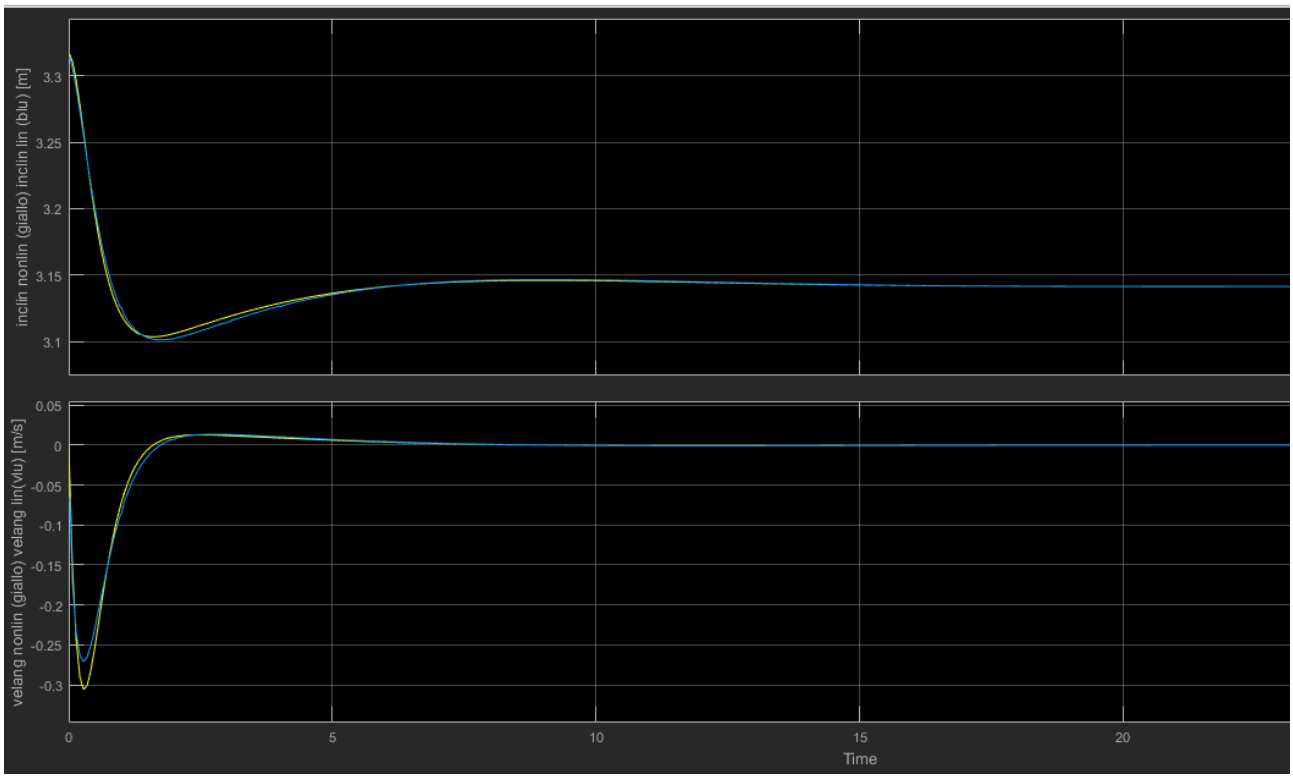
È necessario ora determinare l'istante $\bar{k}$ e quindi si implementa la procedura iterativa che costruisce i vincoli che costituiscono la condizione da valutare e di cui si deve verificare il soddisfacimento.

Durante i primi tentativi si è osservato che il valore di $\bar{k}$ è molto grande perché la procedura di ricerca del $\bar{k}$ impiegava molto a terminare.

Si è deciso perciò di usare un tempo di campionamento pari a 70 ms, sempre dopo aver verificato che questo cambiamento non comprometta il funzionamento del regolatore progettato in precedenza.

Eseguendo questa verifica si ottengono i seguenti andamenti:





La modifica sul tempo di campionamento comporta dei cambiamenti considerati accettabili quindi si è proseguito con la ricerca del $\bar{k}$ che in questo caso risulta circa pari a **480.**

Si può così impostare il problema di programmazione quadratica: come detto prima la funzione obiettivo da minimizzare è in questo caso la norma della differenza tra il riferimento imposto e quello determinato dal Command governor.

È un problema che dovrebbe essere affrontato ad ogni passo di campionamento ma in questa fase si è solo verificato la correttezza del codice.

Definendo come riferimento lo scostamento imposto all'inclinazione del pendolo e come stato iniziale del Command governor la condizione di equilibrio del pendolo si definisce la norma psi e l'insieme di vincoli da passare al comando quadprog, esso restituirà così un riferimento conforme ai vincoli da assegnare al nostro sistema.

```
r = x_scos;
k_segnato = k;
x_cg=[0 0 7/57.3 0]';
psi=eye(length(r));
H=2.*psi;
f2=(-2*(r')*psi)';
A2=[]; b2=[];
for i=0:k_segnato
    [M N]=calcMAT(PHI,i,L,G,Hc,righeR,colonneR);
    A2 = [A2;T*N];
    b2=[b2;g-T*M*x_cg];
end

b2=[b2;g];
temp=T*(Hc*inv(eye(length(PHI))-PHI)*G+L);
A2 = [A2;temp];
w=quadprog(H,f2,A2,b2);
```