POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Controlli Automatici

Levitatore Magnetico

Professore
Stefano Carabelli

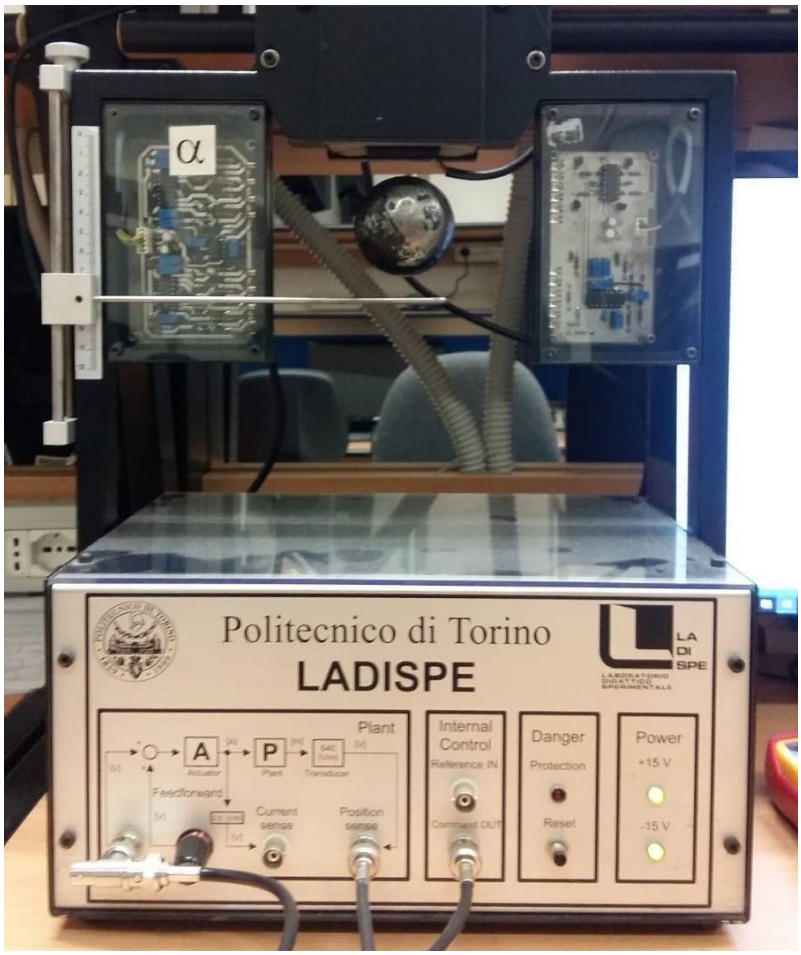
Studenti
Marco Russo
Giulio Pugliese

Anno Accademico 2018/2019

INDICE DEI CONTENUTI

1. Descrizione del sistema

Obiettivo di questo lavoro sperimentale è controllare il levitatore magnetico affinché mantenga una pallina in equilibrio in un punto scelto in fase di progettazione. In particolare, l'equilibrio dovrà essere stabile e la posizione non dovrà cambiare né a seguito di perturbazioni imposte dall'esterno, né a seguito di variazioni sul comando.



Come è possibile vedere in quest'immagine, è stato utilizzato il controllore analogico interno (il BNC va da Command OUT a Command IN) per mostrare il comportamento che vogliamo ottenere.
A livello di impianto, ciò che accade è:
- Vengono sommati il comando "Feedforward" e l'uscita del controllore, entrambi in Volt
- La somma viene amplificata in corrente tramite un **amplificatore di transconduttanza A**
- La corrente risultante va all'**elettromagnete P**, che genera una forza direttamente proporzionale alla corrente e inversamente proporzionale alla posizione (posta crescente dall'alto verso il basso)
- La forza generata comporta un cambiamento di posizione, il cui valore viene portato all'uscita tramite un **trasduttore**.

Notiamo anche una boccola “Current sense”: questa prende l’uscita dell’amplificatore (in corrente) e, tramite un amplificatore di transresistenza (la cui R_m è 2.5 V/A), la converte nuovamente in Volt affinché sia facilmente misurabile. E’ facile presumere che questa transresistenza sia l’inverso della transconduttanza di cui sopra.

Quello che faremo (senza ovviamente utilizzare il controllo interno) è implementare un controllore digitale che prenda in ingresso l’uscita del trasduttore e generi come uscita l’ingresso da dare al Command IN affinché la pallina resti in equilibrio. Osservare che lo scopo non è quindi quello di inseguire un riferimento, ma quello di tenere fissa la pallina nonostante le perturbazioni.

2. Identificazione dei parametri

Essendo la nostra intenzione quella di controllare il sistema del levitatore magnetico, il primo passo da fare è conoscere il sistema che vogliamo controllare. Come descritto sopra, i tre elementi principali dell'impianto sono il trasduttore di posizione, l'amplificatore di transconduttanza e l'elettromagnete. Per ognuno dei tre vi è un'equazione caratteristica (non lineare nel caso dell'elettromagnete) ed occorre trovare i coefficienti di queste equazioni. Non avendo a disposizione dei datasheet, procediamo per via sperimentale: confrontando numerose misure ricaveremo poi, con il metodo dei minimi quadrati, i parametri che cerchiamo.
Va chiarito che tutte le seguenti misure sono riferite al levitatore magnetico "Alpha" messo a disposizione dal Ladispe del Politecnico di Torino.

2.1 Trasduttore di posizione

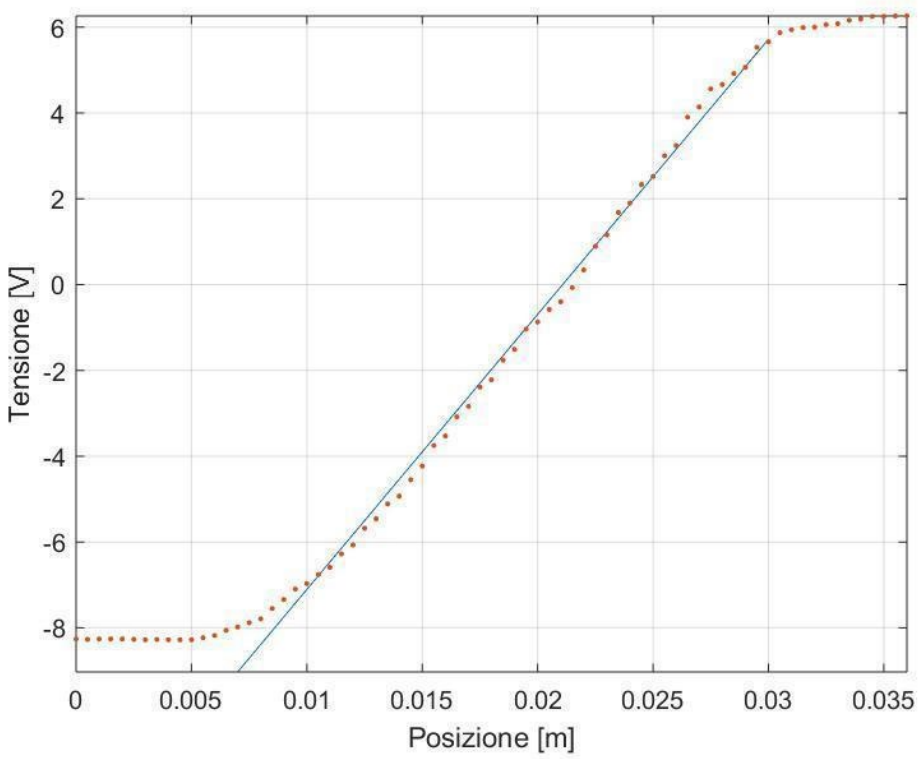
Il levitatore possiede una colonna di fototransistor correlati, dalla parte opposta, a una fila di laser infrarossi. Questa coppia di dispositivi produce una tensione in uscita dalla dinamica piuttosto ampia e particolare rispetto al nostro controllore, sulla boccola "Position Sense". Azzerando il feedforward e il controllo abbiamo misurato, per ogni posizione partendo dalla più vicina all'elettromagnete, la tensione prodotta dal trasduttore.

Il seguente script Matlab genera i coefficienti della retta approssimante la parte lineare del sensore, $v(x) = k_T x + v_0$.
xPos e vPos contengono rispettivamente le posizioni in metri e le tensioni in volt corrispondenti.

```
range=(xm>=0.007&xm<=0.030);      %Prendendo la parte lineare
osservando le misure
a = [xm(range) ones(length(xm(range)),1)] %Aggiunta di vettore
unitario parallelo per ottenere offset
c = vPos(range)
b = a\c    % Utilizzo dei minimi quadrati implementati in Matlab per
generare la pendenza e l'offset

syms t
f = b(1)*t+b(2);     %Retta approssimante
fplot(f,[0.007,0.03]);
hold on
plot(xm,vPos,'.')
xlabel('Posizione [m]');
ylabel('Tensione [V]');
```

Otteniamo b = [641.0988; -13.5159], quindi kT = 641.0988 V/m e v0 = -13.5159 V.
Il trasduttore verrà quindi approssimato sempre con $v(x) = k_T x + v_0$.

Il grafico mostra l'approssimazione e i valori veri delle misure, che agli estremi evidenziano la nonlinearità (di cui non terremo conto, manifestandosi essa solo nelle posizioni estreme e di poco interesse).



Calcolando il valore di posizione x per cui l'uscita è nulla, ossia v(x)=0, si trova x = -v0/kT = 0.0211 m

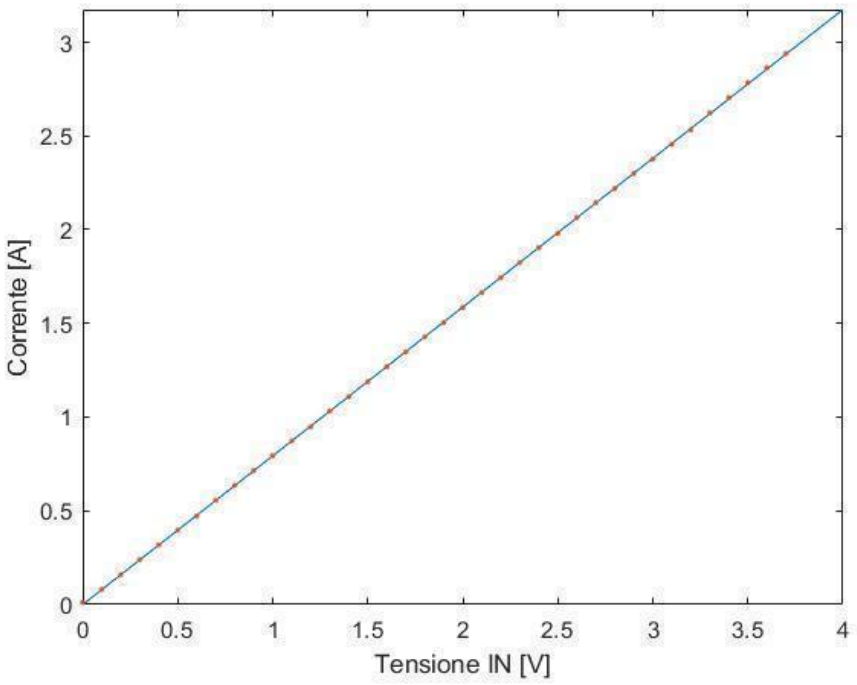
2.2 Amplificatore di transconduttanza

Le misure sono state svolte non sulla corrente ma sulla tensione della boccola "Current Sense". Utilizzando la transresistenza indicata sul levitatore, pari a 2.5 V/A, abbiamo ricavato la corrente corrispondente.

Il seguente script Matlab genera la retta approssimante $i(v) = Gv + i_{0,1}$.
vComm e current sono le nostre misure (quest'ultima è indiretta).

```
a = [vComm ones(length(vComm),1)];    %Aggiunta di vettore unitario
parallelo per ottenere offset
c = current;
b = a\c        %Metodo minimi quadrati

syms t
f = b(1)*t + b(2)     %Retta approssimante
fplot(f,[0,4])
hold on
plot(vComm,current,'.')
xlabel('Tensione IN [V]')
ylabel('Corrente [A]')
```

Otteniamo b = [0.3969; -0.0006], da cui $i(v) = (0.3969\ S)v - 6 * 10^{-4}A$.



E' qualitativamente osservabile la bontà dell'approssimazione.

2.3 Elettromagnete

Per queste misure è stato necessario utilizzare il controllore interno del dispositivo, così da avere una semplice legge che lega la forza magnetica generata con grandezze note.

Collegando "Command out" a "Command in", abbiamo che la pallina è posta all'equilibrio, cioè ferma. Questo significa $F_m = mg$, con $m = 0.022\ kg$. Essendo $F_m = k_m \frac{i^2}{z^2}$, otteniamo $mgz^2 = k_m i^2$. Prevedendo un offset, $mgz^2 = k_m i^2 + i_{0,2}$.

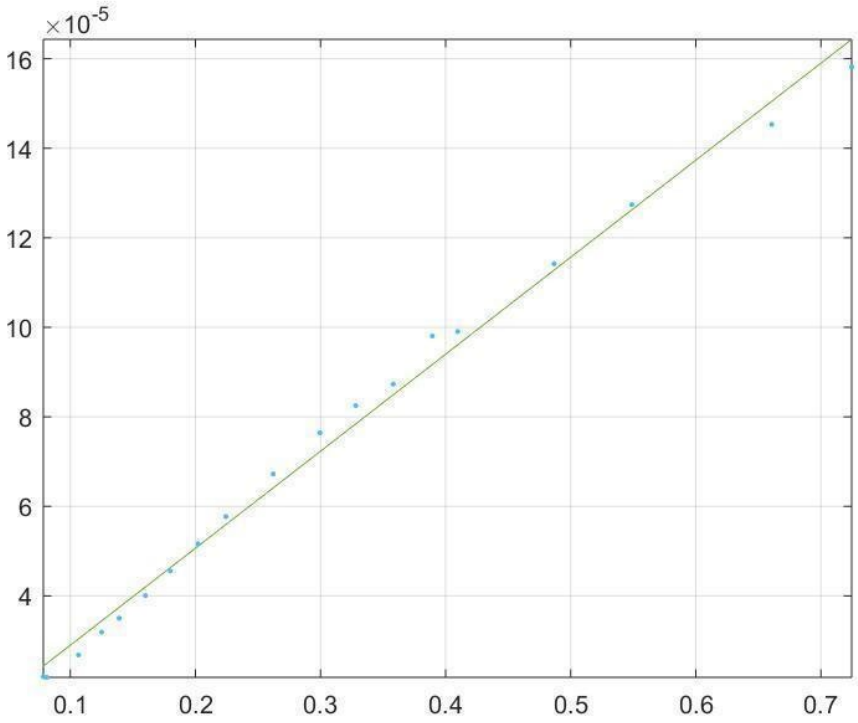
Misuriamo quindi l'uscita del trasduttore di posizione (in volt) e la corrente (misurando prima la tensione di uscita del current sense e poi dividendo per 2.5) facendo variare la posizione della pallina con il feedforward. Questo ci fa ottenere coppie di valori tensione-corrente; essendo già nota l'approssimazione del trasduttore possiamo sfruttarla per ottenere coppie posizione-corrente, completando le grandezze note della relazione.

Le variabili curr e vPos contengono le misure.

```
Vu0 = -13.5159;  %Volt, offset del trasduttore
Kt = 641.0988;  %V/m pendenza del trasduttore
m = 0.022;  %Kg massa della pallina
g = 9.81;  %m/s^2
curr2 = curr.^2;
pos = (vPos-Vu0)/Kt;  %Trasduzione della tensione in posizione
pos2 = pos.^2;      % z^2
A = [curr2, ones(length(curr2),1)];
C = [pos2*m*g];
coeffs = A\C   %Minimi quadrati per ottenere k_m dalla relazione mgz^2 = k_m i^2 + i_0,2
syms t
f = coeffs(1)*t+coeffs(2)  %Retta approssimante, z^2 in funzione dei valori di i^2
fplot(f)
hold on
plot(curr2,m*g*pos2)
```

I calcoli portano ai valori:

$$k_m = 2.17 * 10^{-4}\ \ kg\ m^3 s^{-2} A^{-2}$$
$$i_{0,2} = 7.30 * 10^{-6}\ \ kg\ m^3 s^{-2}$$



i^2 in ascissa, mgz^2 in ordinata.

3. Modellistica

Definiamo x_1 come la posizione e x_2 come la velocità:

$$x_1' = x_2$$
$$mx_2' = mg - F_m$$

Prima, riguardo la forza magnetica, avevamo ottenuto $mgx^2 = i^2 k_m + i_{0,2}$, da cui

$$F_m = \frac{i^2 k_m + i_{0,2}}{x^2} \qquad x_2' = g - \frac{(Gv_{in} + i_{0,1})^2 k_m + i_{0,2}}{m{x_1}^2}$$

In cui siamo passati da corrente a tensione.
L'uscita sarà poi $V_{out} = k_T x_1 + v_0$ (trasduttore) .

Ricapitolando, abbiamo:

- **Variabili di stato:** x1 (posizione), x2 (velocità)
- **Ingressi:** Vin (che poi viene amplificato in i)
- **Uscite:** Vout (trasduttore di posizione)

da cui, in sintesi:

$$x_1' = x_2$$
$$x_2' = g - \frac{(Gv_{in} + i_{0,1})^2 k_m + i_{0,2}}{m{x_1}^2}$$
$$u = V_{in}$$
$$y = k_T x_1 + v_0$$

Osserviamo però che non è ancora possibile ottenere la rappresentazione in matrici di stato, essendo il sistema non lineare: esso andrà quindi linearizzato attorno a un punto di equilibrio.

3.1 Linearizzazione

L'unico termine non lineare è dato dall'accelerazione. Tuttavia, per giungere alla rappresentazione in matrici di stato, è bene considerare l'intero sistema come non lineare: occorre quindi stabilire il punto nominale $\underline{x}, \underline{u}$, dove $\underline{x} = (\underline{x_1}, \underline{x_2})$. La posizione di equilibrio $\underline{x_1}$verrà calcolata come quella per cui l'uscita del trasduttore è 0 V, mentre la velocità di equilibrio $\underline{x_2}$è chiaramente 0.
L'uscita di equilibrio sarà, di conseguenza, $\underline{y} = k_T \underline{x_1} + v_0$.
L'ingresso di equilibrio, infine, sarà una tensione tale che valga $\underline{x} = (0,0)$ (dipenderà dalla posizione di equilibrio scelta).
E' facile notare che, se vogliamo $\underline{y} = 0$ all'equilibrio, dovrà essere $\underline{x_1} = -v_0/k_T$.

Inoltre, dovendo valere $x_2' = g - \frac{i^2 k_m + i_{0,2}}{m x_1^2} = 0$ all'equilibrio, si ottiene $\underline{i} = \sqrt{\frac{g m x_1^2 - i_{0,2}}{k_m}}$.

Essendo $i = G v_{in} + i_{0,1}$, si ha poi $\underline{u} = \left(\sqrt{\frac{g m x_1^2 - i_{0,2}}{k_m}} - i_{0,1}\right) / G$.

Stabiliti i punti, l'ingresso e le uscite di equilibrio, si ottiene

$$\delta x_1' = \frac{\partial f_1}{\partial x_1}|_{\underline{x_1}} \delta x_1 + \frac{\partial f_1}{\partial x_2}|_{\underline{x_2}} \delta x_2 + \frac{\partial f_1}{\delta u}|_{\underline{u}} \delta u \qquad \delta x_2' = \frac{\partial f_2}{\partial x_1}|_{\underline{x_1}} \delta x_1 + \frac{\partial f_2}{\partial x_2}|_{\underline{x_2}} \delta x_2 + \frac{\partial f_2}{\delta u}|_{\underline{u}} \delta u$$
$$\delta y = \frac{\partial g}{\partial x_1}|_{\underline{x_1}} \delta x_1 + \frac{\partial g}{\partial x_2}|_{\underline{x_2}} \delta x_2 + \frac{\partial g}{\delta u}|_{\underline{u}} \delta u$$

ricordando che

$$f_1(x_1, x_2, u) = x_2$$
$$f_2(x_1, x_2, u) = g_k - \frac{(Gu + i_{0,1})^2 k_m + i_{0,2}}{m x_1^2}$$
$$g\,(x_1, x_2, u) = k_T x_1 + v_0$$

e definendo

$$A = \left[\frac{\partial f_1}{\partial x_1}|_{\underline{x_1}} \frac{\partial f_1}{\partial x_2}|_{\underline{x_2}} \frac{\partial f_2}{\partial x_1}|_{\underline{x_1}} \frac{\partial f_2}{\partial x_2}|_{\underline{x_2}}\right] \quad B = \left[\frac{\partial f_1}{\delta u}|_{\underline{u}} \frac{\partial f_2}{\delta u}|_{\underline{u}}\right],$$
$$C = \left[\frac{\partial g}{\partial x_1}|_{\underline{x_1}} \frac{\partial g}{\partial x_2}|_{\underline{x_2}}\right] \quad D = \frac{\partial g}{\delta u}|_{\underline{u}}$$

si ottiene infine

$$[x_1'\ x_2'\,] = [0\ 0\,] + A[x_1\ x_2\,] + Bu$$
$$y = 0 + C[x_1\ x_2\,] + Du$$
$$A = \left[0\ 1\ ;\ 2\frac{(G\underline{u} + i_{0,1})^2 k_m + i_{0,2}}{m x_1^3}\ 0\right] \quad B = \left[0\ ;\ -k_m \frac{2G^2\underline{u} + 2G i_{0,1}}{m x_1^2}\right] \quad C = [k_T\ 0\,] \quad D = 0$$

Ricordando i valori ottenuti nella fase di identificazione:

$$G = 0.3969\ S$$
$$i_{0,1} = -6 * 10^{-4}\ A$$
$$k_m = 2.17 * 10^{-4}\ kg\ m^3 s^{-2} A^{-2}$$
$$i_{0,2} = 7.30 * 10^{-6}\ kg\ m^3 s^{-2}$$
$$k_T = 641.0988\ V m^{-1}$$
$$v_0 = -13.5159\ V$$

Si ottengono i valori di stati e ingresso di equilibrio:

$$\underline{x_1} = 0.0211\ m$$
$$\underline{x_2} = 0\ m s^{-1}$$
$$\underline{u} = 1.61\ V$$

Per verifica, osserviamo che il comando

```
g - (((G*u_eq + i_01)^2)*k_m +i_02)/(m*(x1_eq^2));
```

produce come risultato 0, il che verifica come $f_2\left(\underline{x_1}, \underline{x_2}, \underline{u}\right) = 0$.

Infine si arriva ai valori delle matrici di stato del sistema linearizzato:

$A = [0\ ,1\ ;\ 930.6342,0\ ]$ $B = [0\ ;\ -11.2501\ ]$ $C = [641.0988\ ,0\ ]$ $D = 0$

Col comando eig(A) otteniamo $(30.5063, -30.5063)$, il che dimostra che il sistema è instabile.

4. Progetto del regolatore

Avendo ora a disposizione un modello lineare in rappresentazione di spazio degli stati conoscendone le matrici di stato si può progettare il controllore a tempo continuo, poi discretizzato.
Le specifiche da rispettare sono le seguenti:

- Stabilità asintotica al punto di equilibrio, per variazioni degli stati
- Nessun riferimento da seguire
- Reiezione del disturbo costante in ingresso
- Discretizzazione con periodo di campionamento opportuno

Il regolatore scelto per queste specifiche è un regolatore con piazzamento dei poli e con osservazione degli stati, poichè abbiamo accesso ad uno solo di questi ultimi.

I passi seguiti sono i seguenti, successivamente spiegati nel dettaglio:

1. Verifica di raggiungibilità e osservabilità
2. Imposizione di poli stabili (regolazione statica)
3. Imposizione dei poli dell'osservatore (regolazione dinamica)
4. Integrazione delle variazioni del comando all'ingresso
5. Conversione del regolatore in discreto

Dopo i passi 2,3,4 si ottiene un modello di regolatore che è stato testato su Matlab per verificare le specifiche (soprattutto il non rispetto di esse, per i primi due).
I risultati dei punti 4 e 5 sono stati simulati in Simulink.

Questi sono i parametri e i punti di equilibrio ricavati dall'identificazione:

```
G=0.3969;
i_01= -6*10^(-4);
k_m=0.2167*10^(-3);
i_02=0.0073*10^(-3);
k_T=641.0988;
v_0=-13.5159;
g = 9.81;
x1_eq = -v_0/k_T; %0.0211
m = 0.022;
u_eq = (sqrt((g*m*(x1_eq^2)-i_02)/k_m)-i_01)/G; %1.6128
```

Creiamo le matrici A, B, C e D:

```
A = [0 1; 2*((G*u_eq+i_01)^2*k_m+i_02)/(m*x1_eq^3) 0]
```

```
A = 2×2
         0    1.0000
  930.6342         0
```

```
B = [0; -k_m*(2*G^2*u_eq+2*G*i_01)/(m*x1_eq^2)]
```

```
B = 2×1
         0
  -11.2501
```

```
C = [k_T 0]
```

```
C = 1×2
  641.0988         0
```

```
D = 0
```

```
D = 0
```

Costruiamo il sistema nello spazio degli stati e verifichiamo che sia osservabile e controllabile:

```
sys=ss(A,B,C,D);
tf_sys = tf(sys)
```

```
tf_sys =

           -7212
  -------------------------
  s^2 - 3.553e-15 s - 930.6

Continuous-time transfer function.
```

```
rank(ctrb(sys))
```

```
ans = 2
```

```
rank(obsv(sys))
```

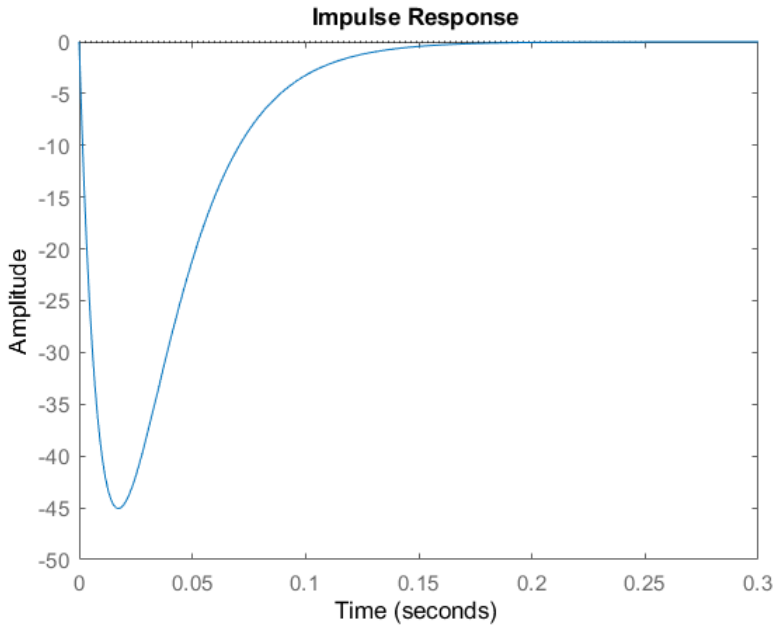
```
ans = 2
```

Essendo sia controllabile che osservabile, possiamo procedere alla retroazione.
Vogliamo una legge di controllo tale da avere un polo in -80 e uno in -40:

```
poles_ctrb=[-80,-40];
K=place(A,B,poles_ctrb)
```

```
K = 1×2
 -367.1646  -10.6666
```

```
A_CL=A-B*K;
sys2=ss(A_CL,B,C,D);
impulse(sys2)
```



Non potendo misurare direttamente gli stati (in realtà non possiamo misurare solo la velocità), costruiamo lo stimatore, imponendo dei poli 10 volte più veloci di quelli appena calcolati:

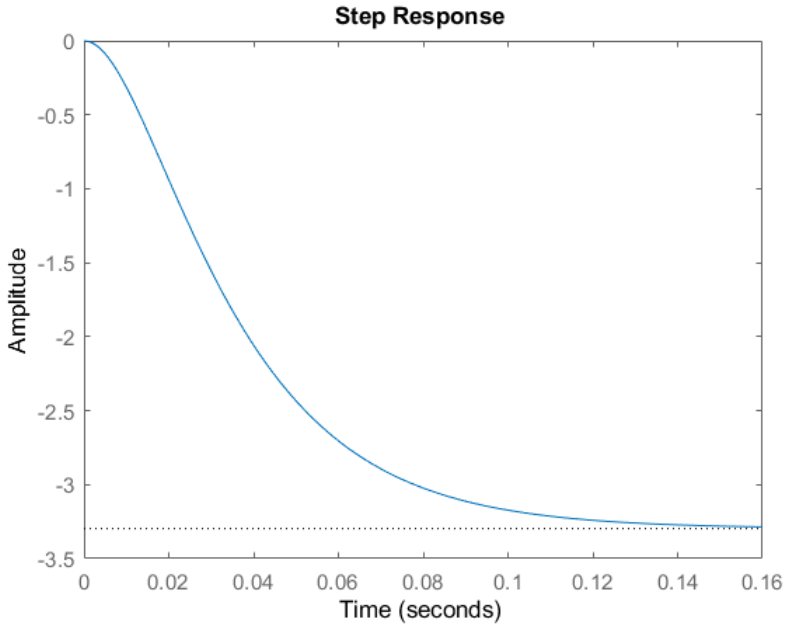
```
poles_obs=poles_ctrb*10;
L = place(A',C',poles_obs)'
```

```
L = 2×1
    1.8718
  500.5947
```

```
A_R = [A-B*K-L*C];
B_R = L;
C_R = -K;
D_R=0;
R = ss(A_R,B_R,C_R,D_R);
S_R=feedback(sys,R,+1); %feedback positivo
impulse(S_R)
```

Il sistema R è stato costruito prendendo come ingresso l'uscita del trasduttore, e come uscita il comando da aggiungersi al feedforward.
Osserviamo però che qui si è assunto come ingresso quello di equilibrio (che quindi per il sistema linearizzato sarà 0) mentre ogni variazione da quell'ingresso è vista come disturbo da rigettare (infatti non diamo le variazioni in ingresso al regolatore). Tuttavia, non siamo ancora in grado di rigettarle: provando, infatti, ad imporre una variazione pari a 1 V:

```
step(S_R)
```



Abbiamo infatti un errore di circa -3.25 V. Piuttosto che dare in ingresso le variazioni del riferimento al regolatore, possiamo aggiungere un controllo integrativo: agli stati di posizione e velocità, va aggiunto quello dell'integrale dell'errore di y-y_des = y-0 = y.
Va da sè che x3' sarà proprio y (la derivata dell'integrale di y).
Osservare che questo stato va aggiunto solo al regolatore; compito di quest'ultimo sarà far tendere questo integrale a 0 (supponendo di imporre poli stabili).
OSS Dato che l'uscita del sistema è la posizione in Volt, x3'=kT*x1.

```
A2 = [A [0; 0]; k_T 0 0];
% A11   A12     0
% A12   A22     0
% k_T   0       0
```

Gli zeri indicano come x1' ed x2' non dipendano da x3, e x3' dipenda solo da x1.

```
B2=[B;0];
%B1
%B2
%0
```

Lo zero mostra che x3' non dipende dagli ingressi (del sistema, non del regolatore).

```
C2=[C 0];
%C1     C2      0
```

L'uscita (del sistema) non dipende da x3.

```
D2=0;
```

L'uscita non dipende dagli ingressi (del sistema).
Il sistema continua ad essere instabile:

```
eig(A2)
```

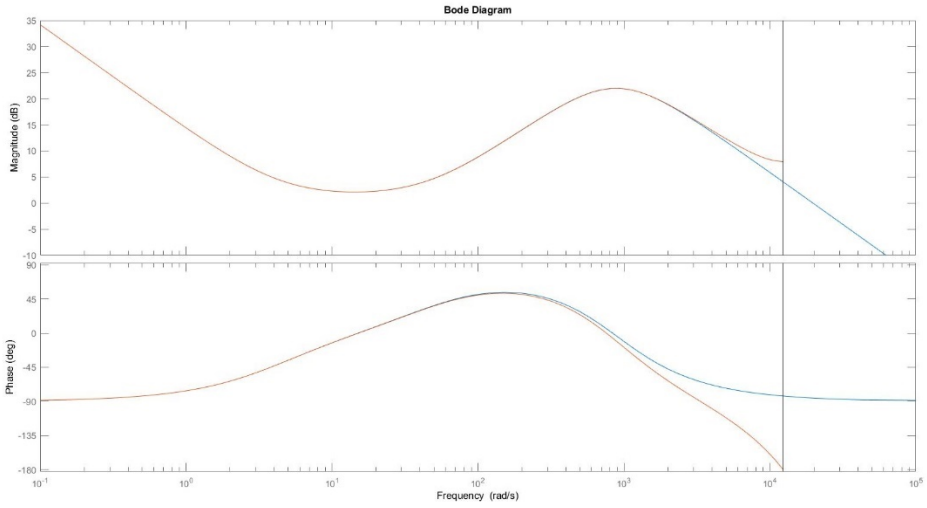
```
ans = 3×1
         0
   30.5063
  -30.5063
```

Per lo stesso discorso di cui sopra, queste matrici servono solo a "modellare" il sistema dal punto di vista del regolatore aumentato. Le matrici di sistema restano quelle dell'inizio!

```
poles_ctrb2=[-300,-60,-5];
K2 = place(A2,B2,poles_ctrb2);
```

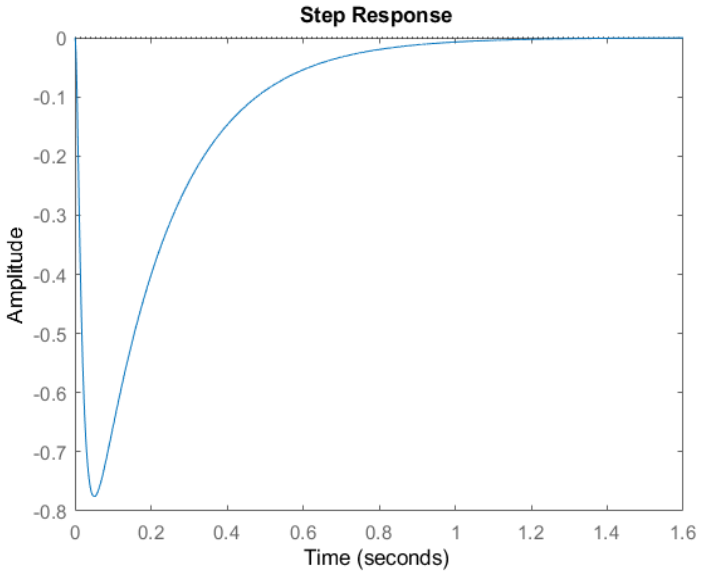
L'uscita del regolatore dovrà stavolta essere -K2*[x1;x2;x3].

```
A_R2 = [A-L*C-B*K2(1:2) -B*K2(3); 0 0 0];
B_R2 = [L;1];
C_R2 = -K2;
D_R2 = 0;
```



Osserviamo come si parta da -20 db/dec: questo è dovuto al polo in 0 relativo all'azione integrativa. Il diagramma di Bode è quello tipico di un PID le cui tre azioni sono tutte applicate all'uscita del sistema: l'azione proporzionale è infatti quella relativa alla posizione stimata, quella integrativa è relativa all'errore (integrale dell'uscita), mentre quella derivativa è relativa alla velocità che è, per l'appunto, la derivata della posizione.

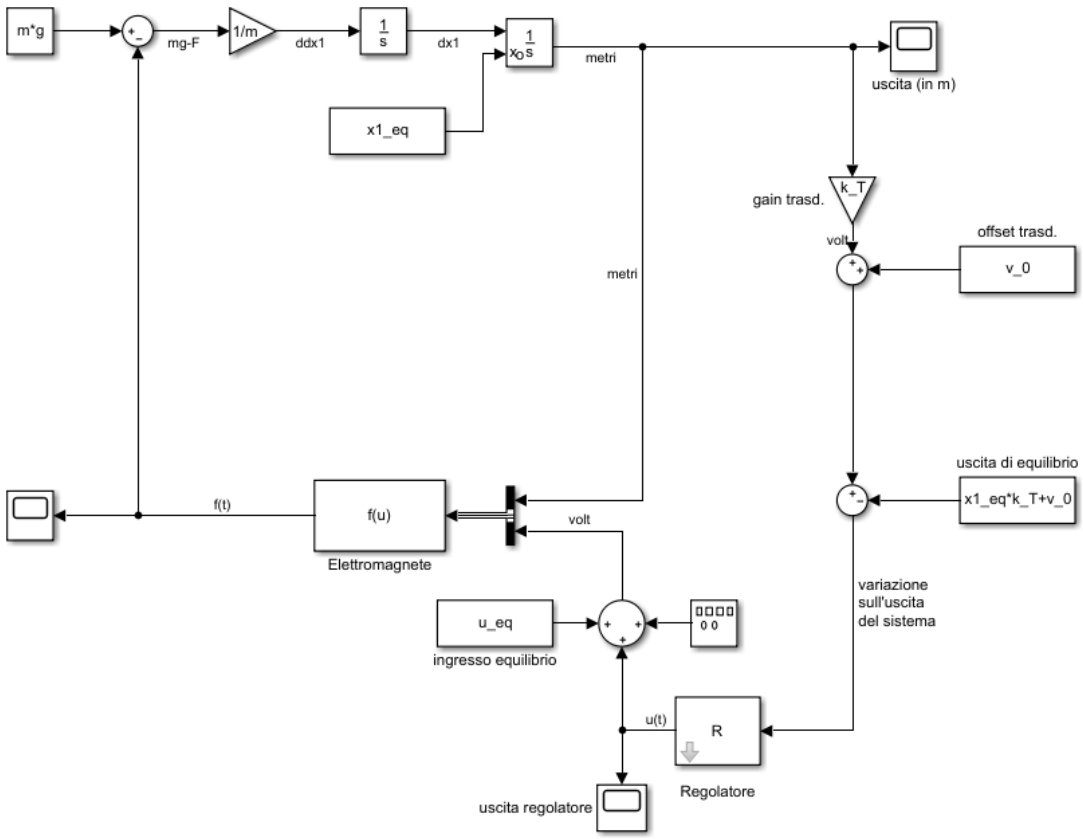
```
R = ss(A_R2,B_R2,C_R2,D_R2);
S_R = feedback(sys,R,+1);
step(S_R)
```



Il risultato è quello voluto: anche a variazioni (contenute) dell'ingresso, la posizione viene riportata all'equilibrio.

4.1 Simulazione

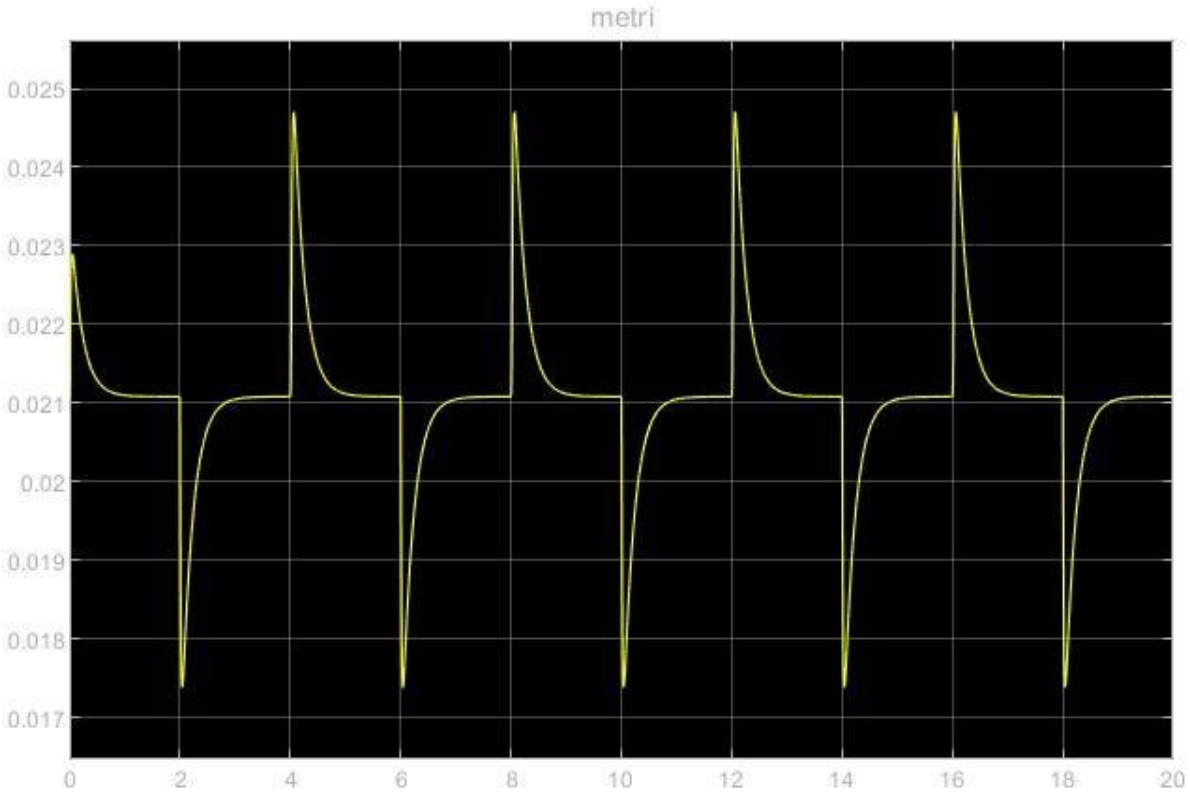
Il modello Simulink con cui simulare è il seguente:



Da notare come l'impianto da controllare sia rimasto non lineare e nel dominio del tempo, mentre il regolatore non riceve in ingresso la "vera" uscita del sistema, ma solo la sua variazione rispetto a quella di equilibrio. Il fatto che queste coincidano è chiaramente solo dovuto al fatto che l'uscita di equilibrio del sistema è 0 V.
Notare inoltre che l'uscita del regolatore, essendo l'ingresso da dare all'impianto, va sommata poi all'ingresso di equilibrio per lo stesso motivo.

Si è aggiunto un generatore ad onde quadre di frequenza 0.25 Hz per simulare una variazione periodica del feedforward di 1.5 V rispetto ai 1.61 V di equilibrio.
Ciò che ci aspettiamo prima di simulare è che la posizione, dopo esser stata perturbata, torni automaticamente all'equilibrio.

Visualizzando lo scope all'uscita del sistema otteniamo infatti quanto segue:



Notiamo in particolare come, anche se il feedforward varia da 0.11 a 3.11 V, la posizione viene perturbata di al massimo 2 mm prima di tornare al punto nominale.

Facendo varie prove si è inoltre osservato che, in linearità, l'uscita del regolatore è contenuta nel range (-3,+3) V, che considereremo pari a (-4,+4) V per abbondare.

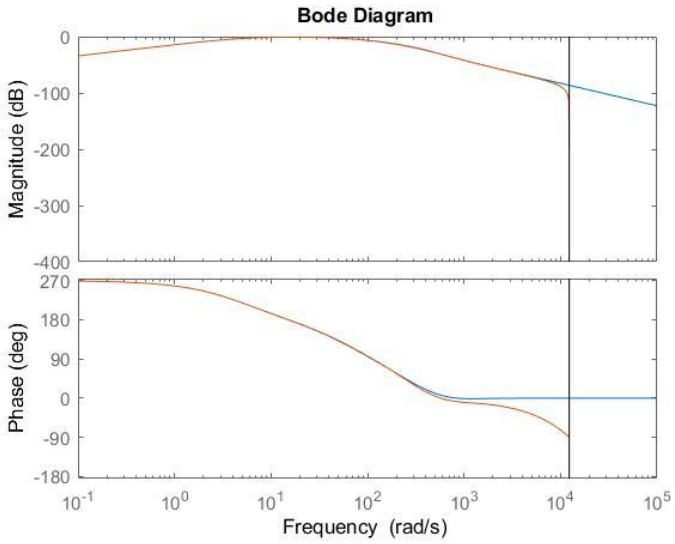
4.2 Discretizzazione

Si procede infine a discretizzare il regolatore progettato e si ripete la simulazione su Simulink, per verificarne l'uguale efficacia.
Poiché Simulink può utilizzare direttamente un sistema ibrido continuo-discreto è sufficiente produrre il regolatore discreto con la funzione c2d, qui utilizzata con il metodo ZOH.

Scegliamo come frequenza di campionamento una pari a quattro volte quella del PWM, quindi 3.92 kHz:

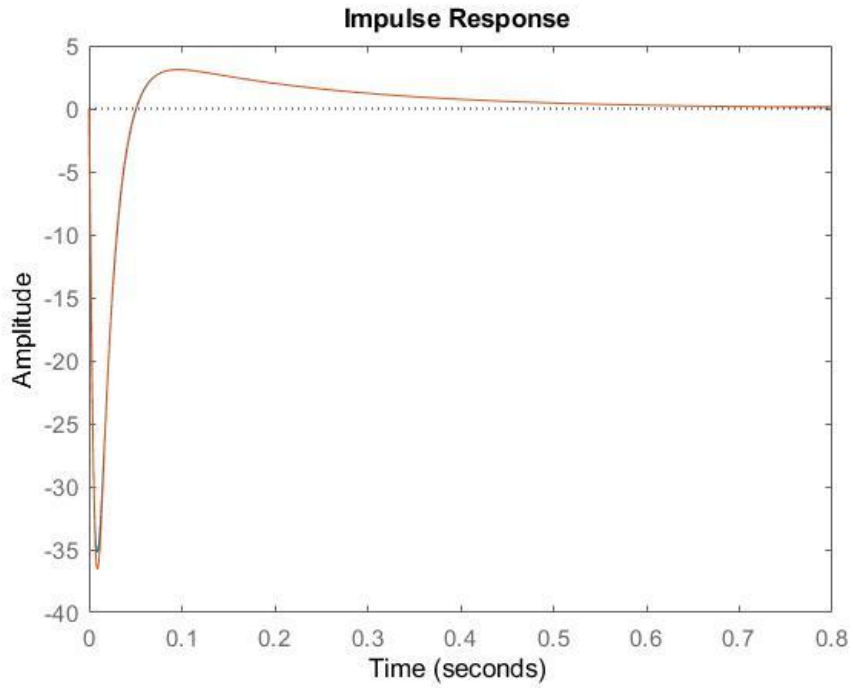
```
R_discr = c2d(R,1/(4*980));
sys_discr = c2d(sys,1/(4*980));

S_R_discr=feedback(sys_discr,R_discr,+1);
```

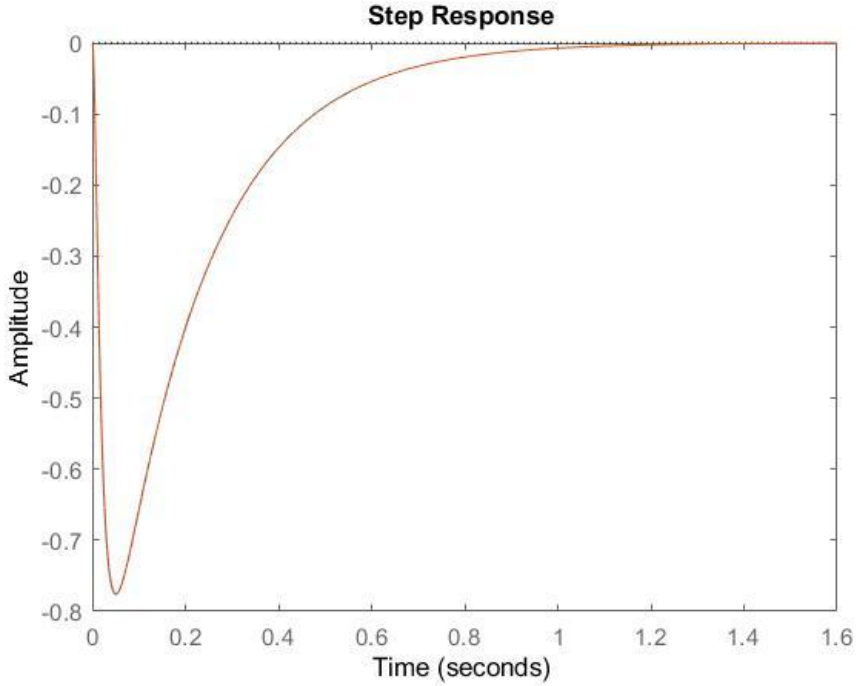
Abbiamo discretizzato separatamente sistema e regolatore in quanto quest'ultimo servirà poi per l'implementazione su Arduino.
Il diagramma di bode del sistema in catena chiusa originario e discretizzato è il seguente:



In blu vi è il sistema originario. Notiamo una sovrapposizione perfetta fino ai 900 rad/s (la distorsione di fase avviene prima di quella di ampiezza).

Visualizzando le risposte all'impulso e al gradino, sovrapponendo la risposta a tempo continuo e tempo discreto, otteniamo:





La sovrapposizione è praticamente perfetta, il che è dovuto all'elevata frequenza di campionamento.

Osserviamo che i poli del nuovo sistema in catena chiusa sono:

9.99e-01
9.85e-01
9.28e-01 + 3.24e-02i
9.28e-01 - 3.24e-02i
7.91e-01

Avendo tutti parte reale in modulo minore di 1, il sistema discretizzato è stabile.
Osserviamo che, nel sistema continuo in catena chiusa, il più grande polo in modulo è pari a circa -800, cui corrisponde una frequenza di circa 127 Hz. Con 3.92 kHz siamo quindi ampiamente sopra la frequenza di Nyquist.

5. Implementazione digitale

Per realizzare il controllo digitale, abbiamo scelto di utilizzare un Arduino Uno.
Le sue caratteristiche principali sono le seguenti:

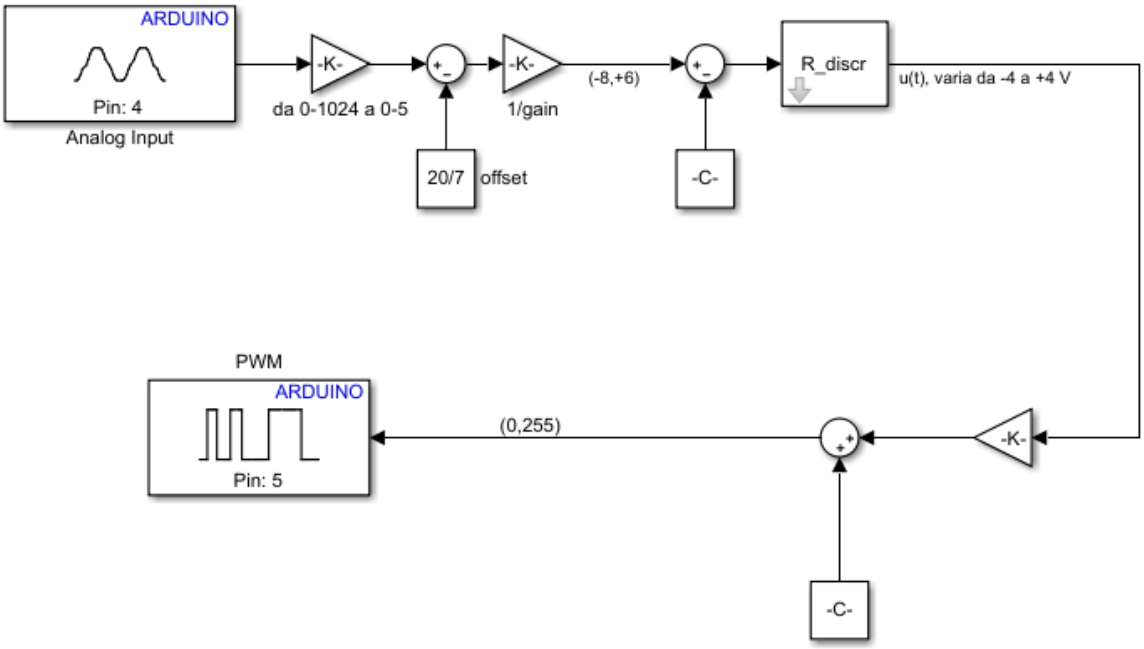
Frequenza massima di campionamento	9.6 KHz
Bit del convertitore A/D	10 bit
Dinamica ingresso e uscita	Tra 0 e 5 V
Uscita in PWM	Tra 0 e 5 V con frequenza di 980 Hz

Utilizzando un pin analogico si affida alla scheda il compito di convertire la tensione letta in un valore digitale su 10 bit, scegliendone la frequenza di campionamento che nel nostro caso da progetto è di **3.92 KHz**. La tensione convertita è quindi riportata internamente come variabile intera tra 0 e 1023 con errore di quantizzazione di $\frac{1}{2}\frac{1}{2^{10}} = 0.05\%$ (relativamente ai 5V di fondo scala).

Il regolatore discreto da noi progettato, tuttavia, non è costruito per ricevere valori interi di questo tipo ma ha come ingresso la tensione del position sense tra -6 e +8 V. Quindi per ricostruire questi valori a partire da numeri su 10 bit si utilizzano semplici guadagni e offset, coincidenti con gli inversi di quelli del circuito di interfacciamento in ingresso descritto nel punto successivo.

Inoltre, Arduino Uno non possiede pin analogici di uscita ma può solamente utilizzare un PWM (Pulse Width Modulation) che correla un numero su 8 bit al valor medio di un'onda quadra di frequenza 980 Hz. Ciò è svolto ponendo come duty cycle dell'onda tra 0 e 5 V proprio la percentuale corrispondente al suddetto numero tra 0 e 255 (cui corrisponde 0-100% di duty cycle).
Il valor medio è $V_m = D * 5$, cioè una frazione dei 5 V.

Come per l'ingresso, anche l'uscita del regolatore discreto va scalata e traslata perchè è progettato per fornire tensioni tra -4 e +4 V e non un numero intero tra 0 e 255.
Di seguito lo schema a blocchi con il toolbox **Simulink Support Package for Arduino Hardware**, che permette di generare il codice di basso livello da impiegare su Arduino utilizzando un approccio model-based (quindi di alto livello). Il modello Simulink è il seguente:



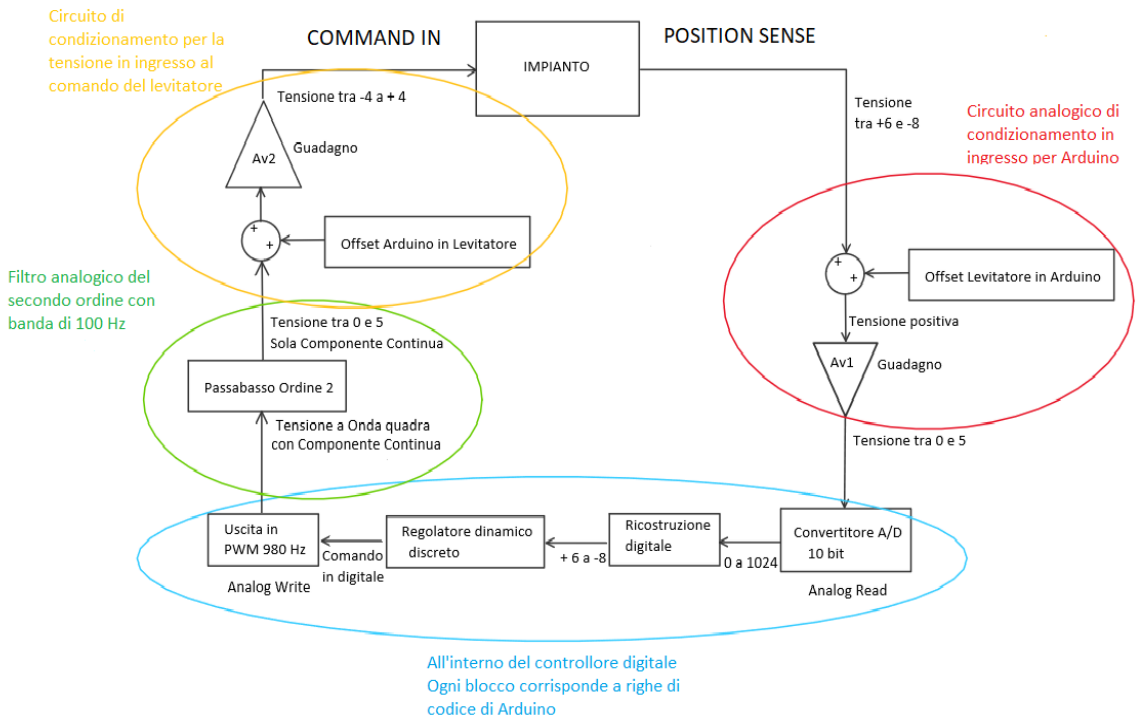
In ingresso si fa in modo che i valori, che variano da 0-1024, tornino nel range (-6,+8). In uscita, invece, si adatta il range (-4,+4) a (0,255).
Utilizzando la funzione “Deploy to Hardware”, dopo aver opportunamente configurato il modello Hardware, Simulink compila automaticamente il relativo codice e lo carica sull’Arduino.
Osservare che, nelle impostazioni di simulazione, il passo deve essere “Fixed” e non “Variable”.

Nella prossima sezione verrà infine discussa la realizzazione elettronica dell’interfacciamento.

6. Circuiti di condizionamento e filtraggio

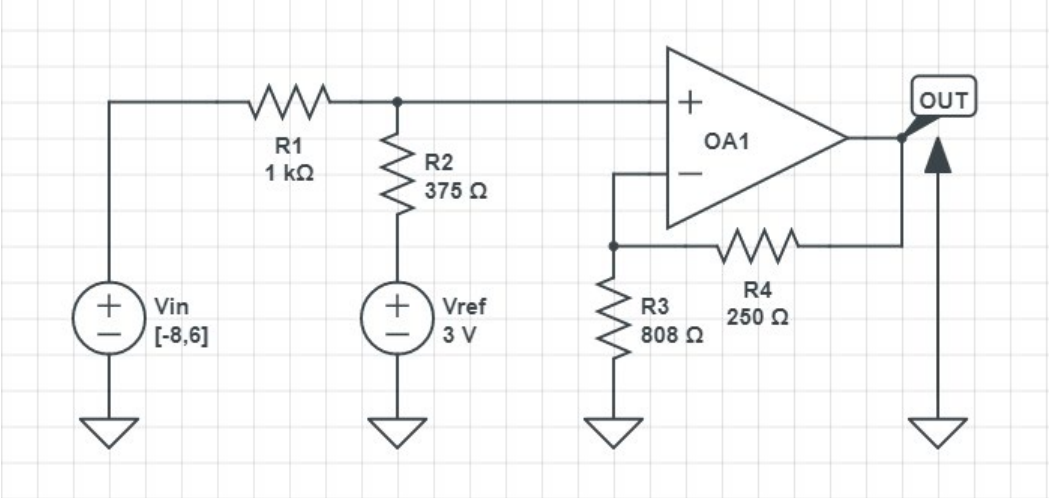
Dovendo dare come input ad Arduino il position sense che varia tra -8 e +6 V, è chiaro che, per non danneggiarlo e per operare correttamente, occorre scalare e traslare questo segnale. Inoltre, va progettato un filtro passabasso per estrarre la componente continua dal segnale in uscita dal PWM, e un ulteriore circuito di condizionamento per portare gli 0/+5 V risultanti ai -4/+4 V effettivi generati dal regolatore.

Lo schema risultante dovrà quindi essere il seguente:



6.1 Input conditioning

Come sopra anticipato, dato che il regolatore prende in ingresso il Position Sense (che ha un range da -8 a +6 V) mentre Arduino Uno supporta tensioni che vanno da 0 a +5 V, occorre progettare un circuito attivo che produca un offset e un gain adatto.
In particolare, vogliamo una $V_{out} = KV_{in} + V_0$. Imponendo che la retta passi per $(-8;\ 0)$ e $(6; 5)$ otteniamo $K = 0.357, V_0 = 2.857\ V$.
Per mezzo di un amplificatore operazionale e un riferimento costante (pari a 3 V) è facile ottenere questo risultato, come dimostra il seguente circuito:

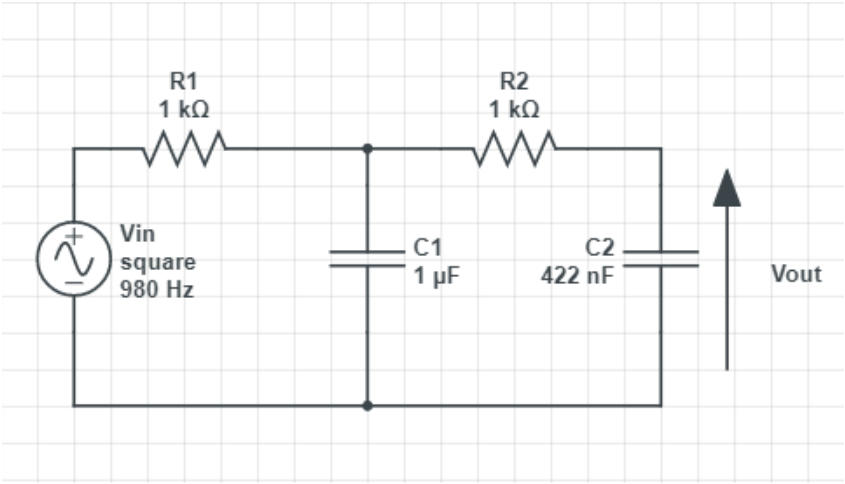


Utilizzando il teorema di Millman, si ottiene una $V_+ = V_- = \frac{V_{in}R_2 + V_{ref}R_1}{R_1 + R_2}$. Inoltre, $V_{out} = V_+ \left(1 + \frac{R_4}{R_3}\right)$. Provando a sostituire i valori, si ottiene il risultato desiderato.

6.2 PWM filtering

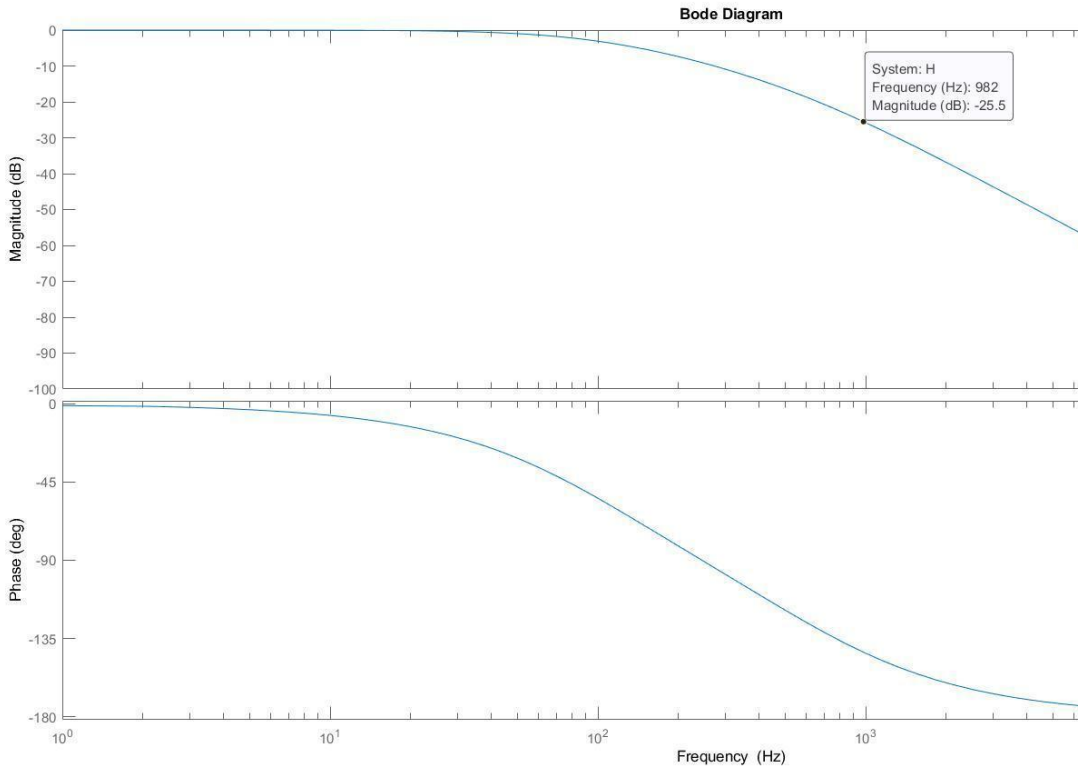
L'output dell'Arduino è costituito da un DAC PWM a 980 Hz, con un range di tensioni da 0 a +5 V. Volendo estrarre la sola componente continua, si potrebbe essere tentati di utilizzare un filtro passa-basso con banda estremamente ristretta. Questo, però, porterebbe a costanti di tempo molto elevate, rendendo la dinamica del filtro più lenta di quella del controllore.

Dalla teoria dei segnali sappiamo però che lo spettro di un PWM è non nullo in f = 0 Hz (DC) e nullo in $(0, f_{pwm})$ (in f_{pwm} e nei suoi multipli, però, l'ampiezza dello spettro non è nulla). Per questo motivo, piuttosto che filtrare a frequenze molto basse, basta filtrare in prossimità dei 980 Hz (il risultato è teoricamente identico).
Notiamo però che il filtro passa-basso ideale (cioè con pendenza infinita) non è fisicamente realizzabile, e possiamo al più costruire filtri con pendenze in AF di $-n\ 6\ dB/oct$, dove n è pari al numero dei poli (o più in generale all'ordine del filtro).
Abbiamo quindi scelto di progettare un filtro passa-basso passivo del secondo ordine siffatto:



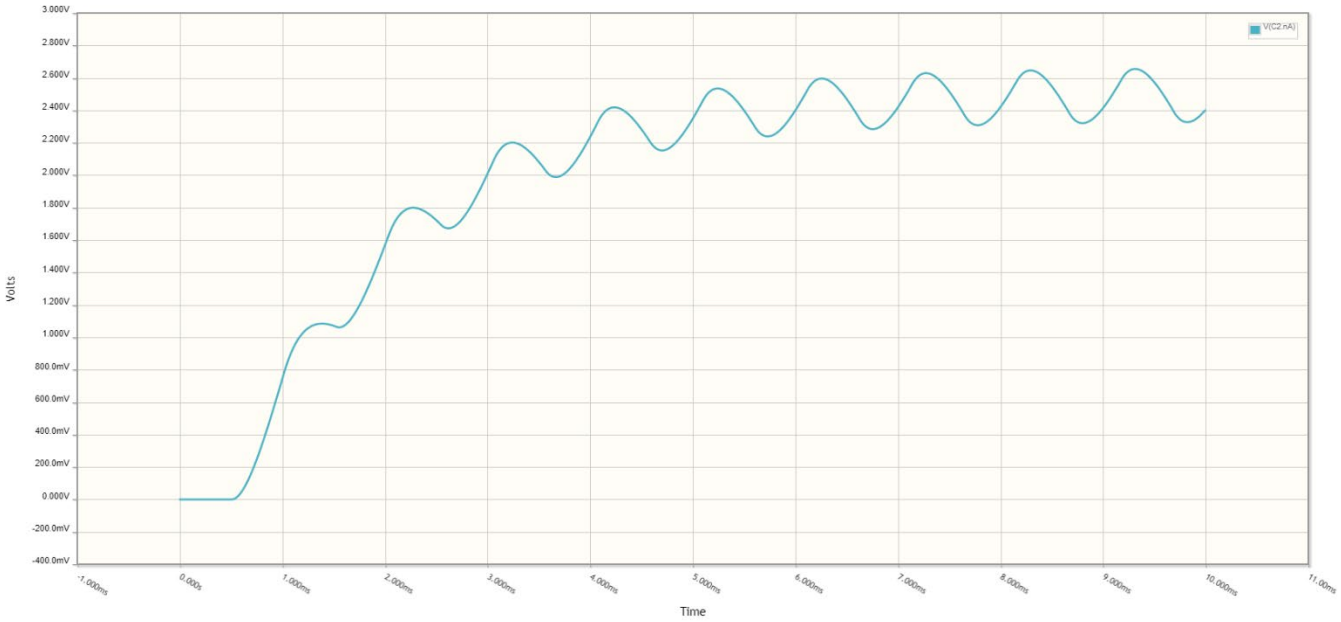
I valori dei componenti sono stati scelti in modo da ottenere -24 dB a 980 Hz (il ripple viene quindi attenuato di 15.8) e una banda passante di circa 100 Hz.

La funzione di trasferimento risultante è $H(s) = \frac{2.37*10^6}{(s+3736)\,(s+634.4)}$, da cui otteniamo il seguente diagramma di Bode:



mentre i poli non nulli sono -634.4 rad/s e -3740 rad/s (rispettivamente 101 Hz e 596 Hz. Notiamo che il più piccolo (in modulo) è circa due volte più grande del più grande (in modulo) dei poli del regolatore.

Provando a simulare con CircuitLab ciò che accade quando il valore da convertire è 2.5 V:

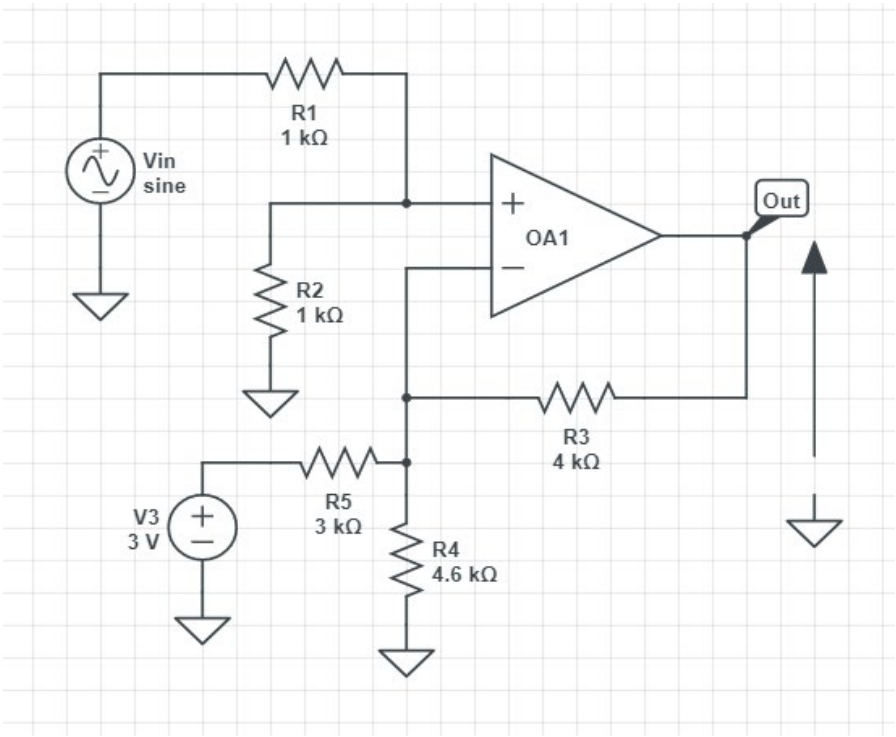


E’ possibile vedere che il segnale oscilla, dopo la salita (circa 5 ms), tra 2.6 e 2.4, avendo quindi un ripple ampio 100 mV, che è comunque un buon risultato avendo utilizzato un filtro del secondo ordine (per utilizzi più critici sarebbe idoneo progettare filtri più avanzati, come Butterworth, Čebyšëv, ecc…, che, a fronte di una maggiore complessità circuitale, possono offrire pendenze più elevate).

Va precisato che, nel caso reale, le variazioni del Duty Cycle portano alla comparsa di componenti spettrali alle subarmoniche: ipotizzando che il DC vari poco, tuttavia, è possibile trascurare queste componenti, influendo queste relativamente poco sul ripple.

6.3 Output conditioning

L'ultimo circuito da progettare è quello che deve condizionare il segnale filtrato post-PWM da 0/+5 V a -4/+4 V (su Simulink è stata osservata questa dinamica di uscita del regolatore). Il circuito, simile al primo, è il seguente:

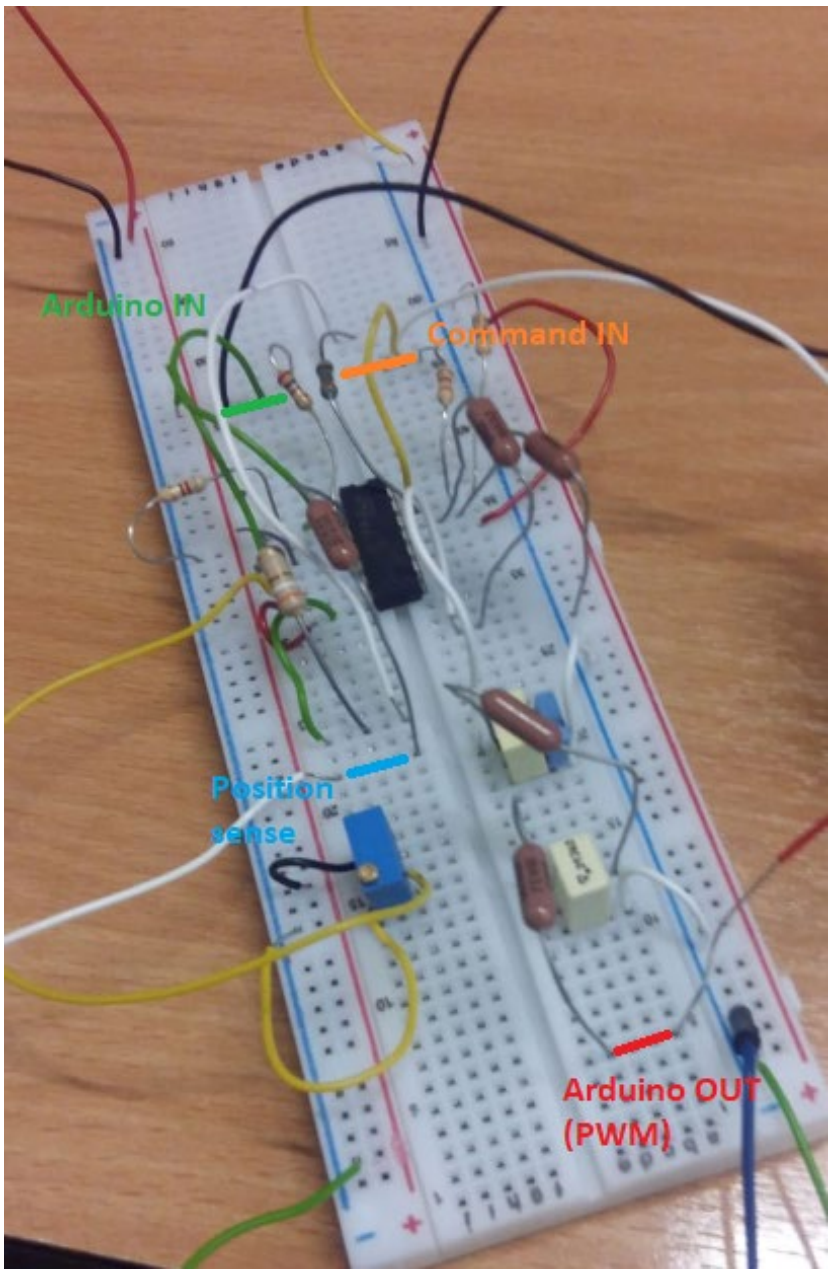


6.4 Realizzazione

Per realizzare questi circuiti è stato utilizzato:
- un integrato ST TL084CN, contenente quattro operazionali
- un alimentatore duale master-slave Topward TPS-4000 (alimentando l'integrato da -10 a +10 V)
- una breadboard
- appositi condensatori e resistenze (ovviamente non disponibili con i valori esatti sopra presentati, restando comunque in tolleranze accettabili)
- un trimmer per ottenere 3 V a partire dai 10 V di alimentazione

Inoltre, per il testing:
- un generatore di funzioni HP 33120
- un oscilloscopio digitale Tektronix TDS310
- un multimetro palmare

Il risultato è il seguente:

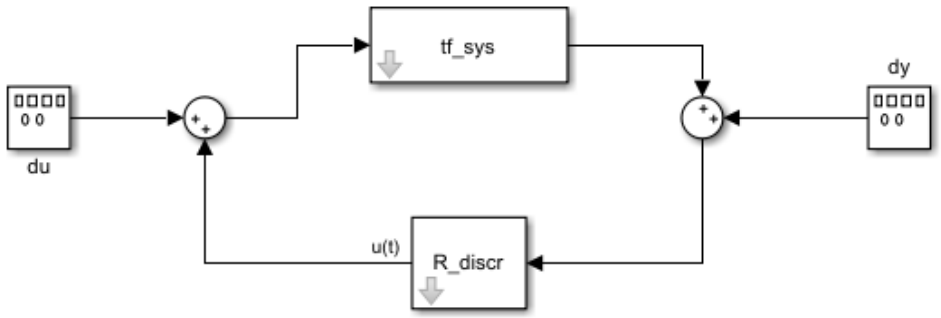


A sinistra vi è l'input conditioning, in basso a destra il filtro post-PWM e in alto a destra l'output conditioning.

7. Reiezione dei disturbi

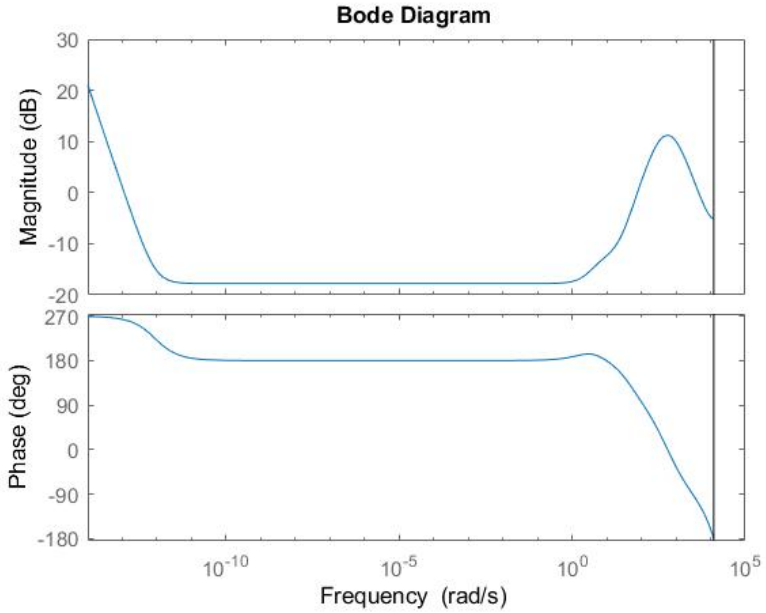
Prima di effettuare una simulazione completa del PWM (che non riusciamo a modellare efficacemente in catena chiusa con delle funzioni di trasferimento) possiamo fare delle stime sulla reiezione dei disturbi sul comando e sull'uscita del sistema a tempo discreto, privo per l'appunto della parte PWM-filtro passa basso (a causa di ciò dovremo fare delle simulazioni complete di queste reiezioni nel capitolo successivo).

Il sistema linearizzato e con i disturbi è il seguente (tf_sys è qui discretizzato):

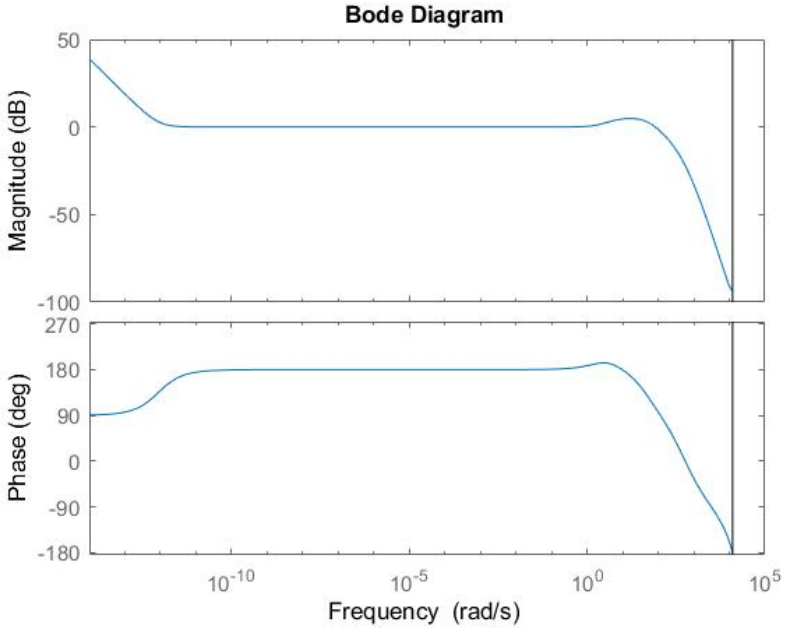


E' facile calcolare che:

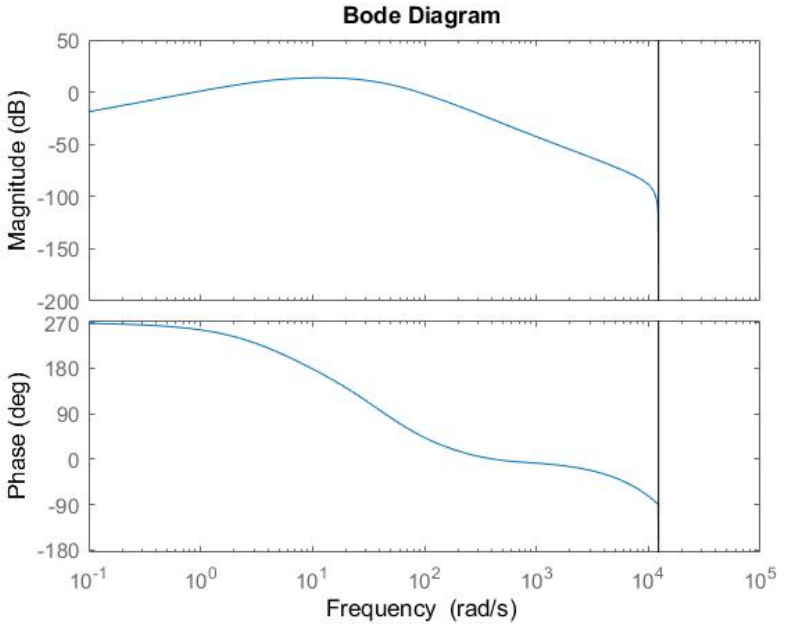
$$\frac{U(s)}{d_y(s)} = \frac{R_{discr}}{1 - R_{discr} tf_{sys}}$$



$$\frac{Y(s)}{d_y(s)} = \frac{R_{discr}tf_{sys}}{1 - R_{discr}tf_{sys}} = \frac{U(s)}{d_{u(s)}}$$



$$\frac{Y(s)}{d_u(s)} = \frac{tf_{sys}}{1 - R_{discr}tf_{sys}}$$



Da queste informazioni, è possibile stabilire che:

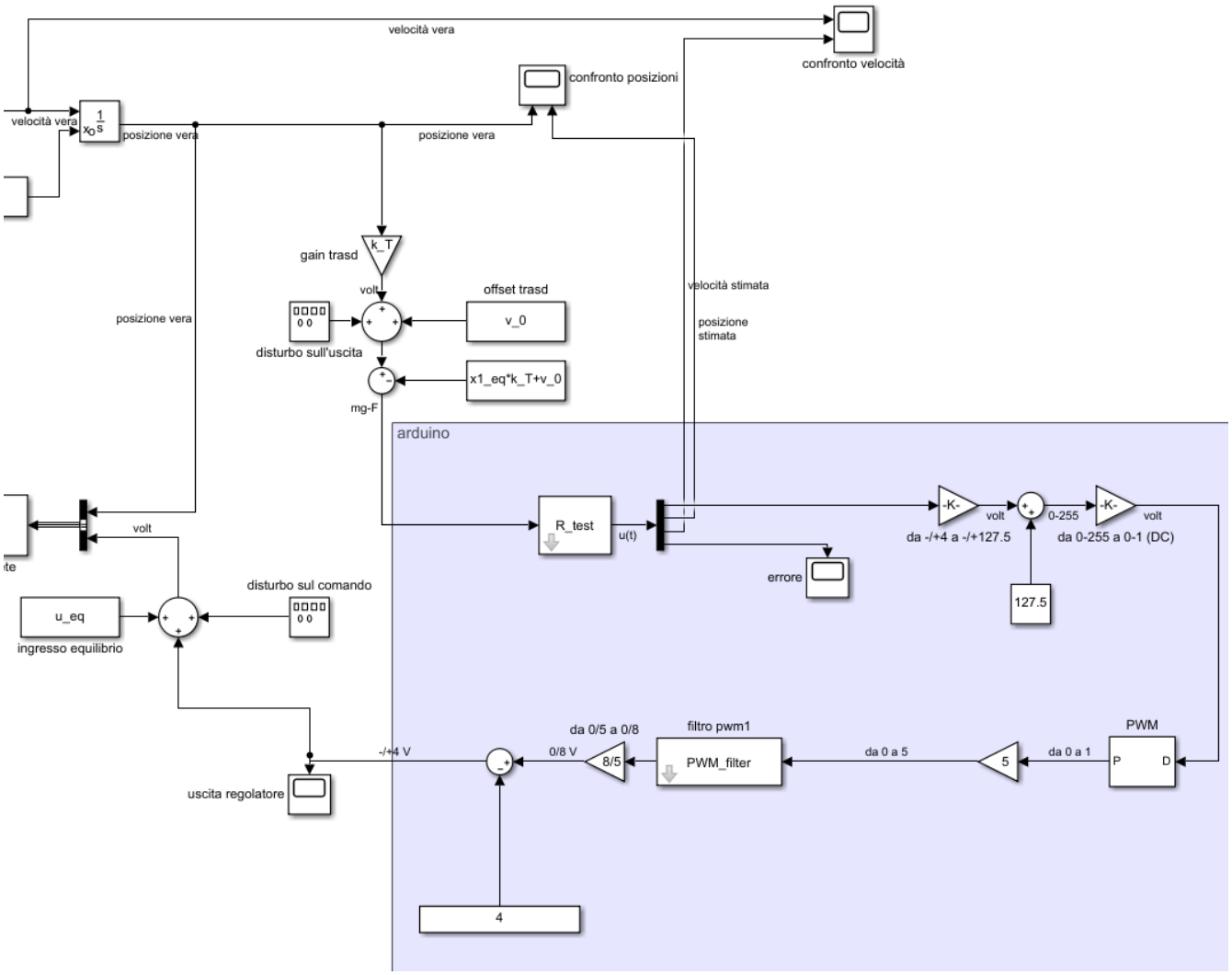
- prendendo come output il comando e come input il disturbo sull'uscita, il disturbo viene attenuato di circa -18dB da 10^-12 a 1 rad/s, mentre viene amplificato fuori da questa banda
- prendendo come output il comando e come input il disturbo sul comando, oppure come output l'uscita e come input il disturbo sull'uscita, il disturbo viene amplificato fino a 10^-12, passa da 10^-12 a 1 rad/s e viene attenuato da 1 rad/s in poi
- prendendo come output l'uscita e come input il disturbo sul comando, il disturbo passa da 0.6 a 100 rad/s circa, mentre viene attenuato fuori da questa banda.

8. Simulazione completa

Volendo tenere conto del PWM e del relativo filtro, abbiamo eseguito una simulazione completa di queste dinamiche. In particolare, abbiamo modificato leggermente la matrice di uscita del regolatore in modo da ottenere, oltre al comando, anche la posizione e la velocità stimate e l'integrale dell'errore:

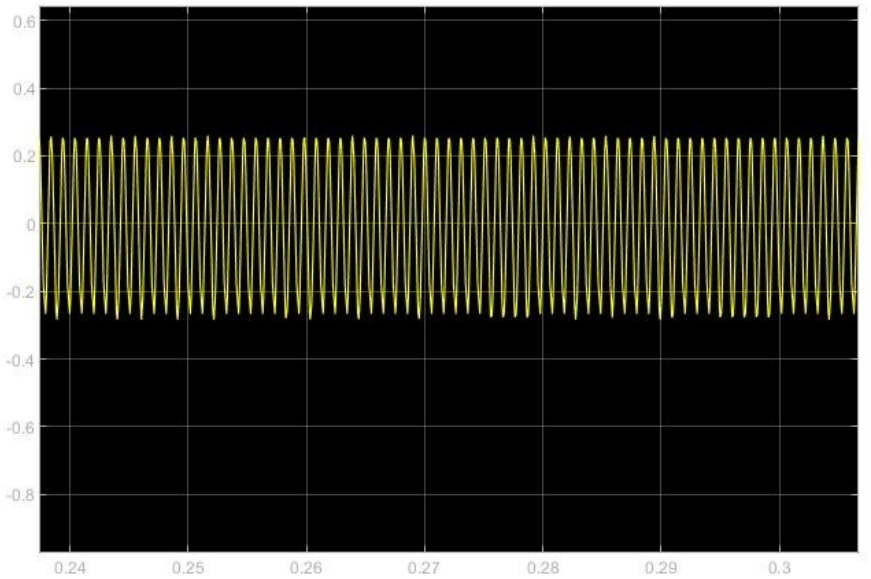
```
PWM_filter=2.37*10^6/((s+3736)*(s+634.4));
C_R3=[-K2; [1 0 0]; [0 1 0]; [0 0 1]];
D_R3=[0;0;0;0];
R_test = ss(A_R2,B_R2,C_R3,D_R3);
```

Per motivi di spazio, riportiamo solo la parte di schema che abbiamo modificato:

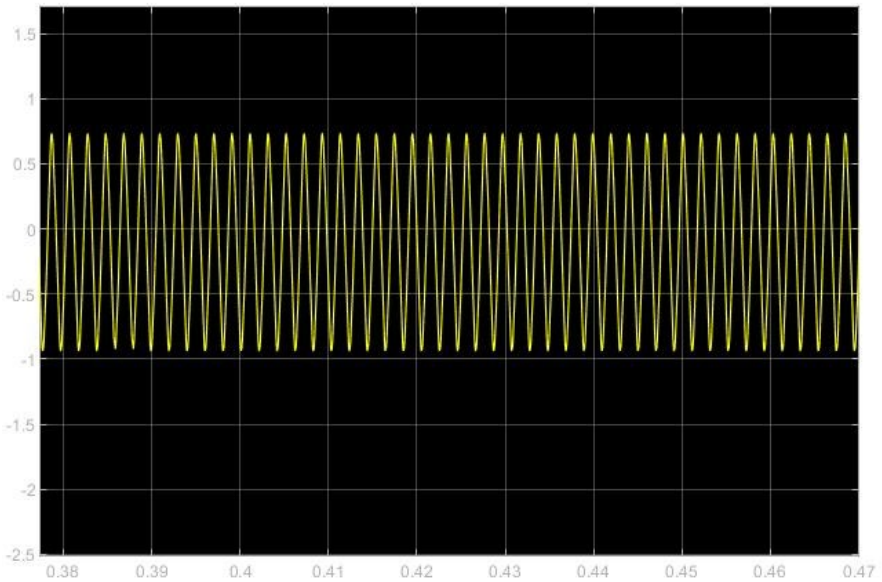


Il blocco PWM è preso dalle librerie “Simscape Electrical”.
Abbiamo incluso per completezza gli scalamenti che avvengono nell’Arduino all’uscita del regolatore, e anche dopo il filtro passa basso. Sono stati invece trascurati gli scalamenti dell’input conditioning visto che poi vengono ricompensati nell’Arduino stesso.

Testiamo innanzitutto la quantità di ripple dovuta al PWM:

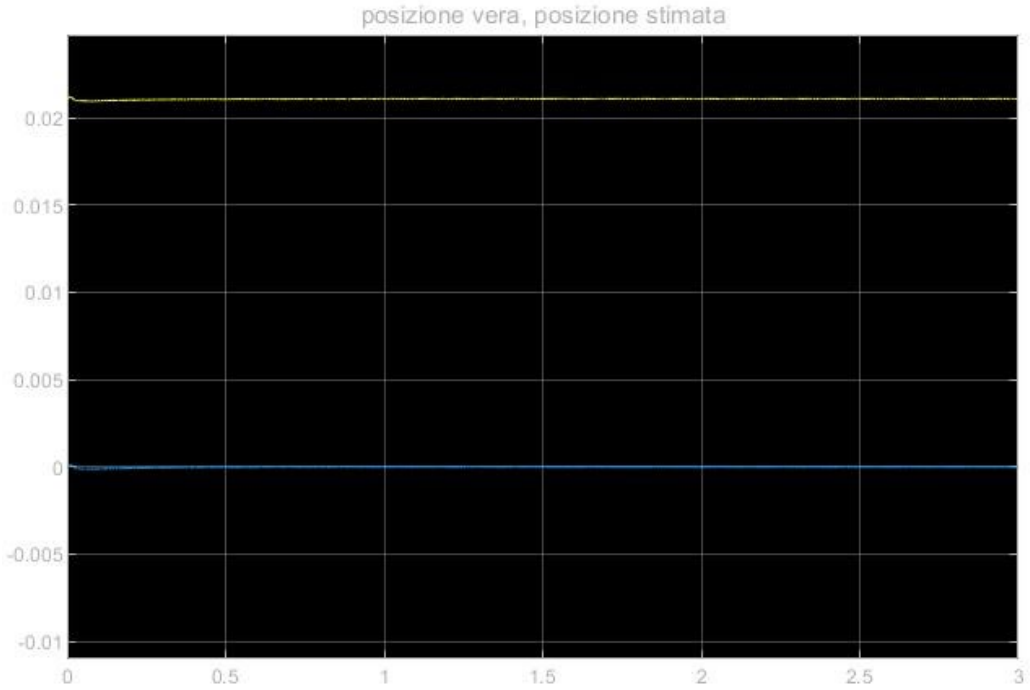


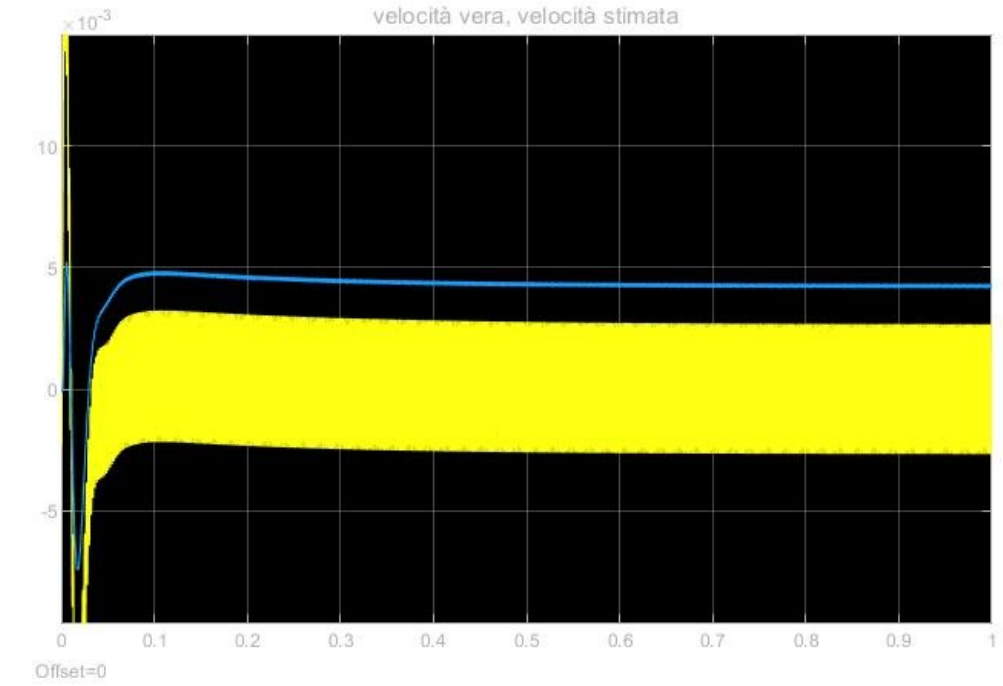
A 980 Hz abbiamo, come previsto, un ripple ampio circa 220 mV.
Possiamo a questo punto chiederci quanto influisce l’attenuazione del filtro passa basso sull’ampiezza del ripple: piuttosto che modificare la funzione di trasferimento del passa basso, abbassiamo la frequenza del PWM così da avere una minore attenuazione: con il PWM a 490 Hz, il risultato che otteniamo è questo:



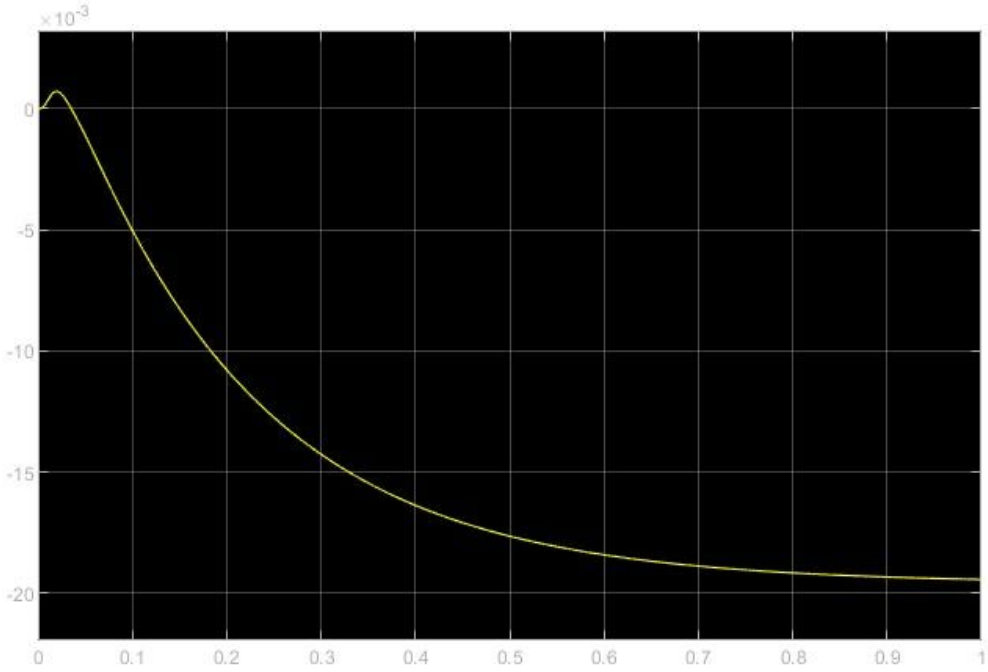
Abbiamo quindi un ripple ampio circa 0.75 V, più di tre volte rispetto a prima.

Possiamo ora confrontare posizioni e velocità vere (giallo) e stimate (blu) in assenza di disturbi:

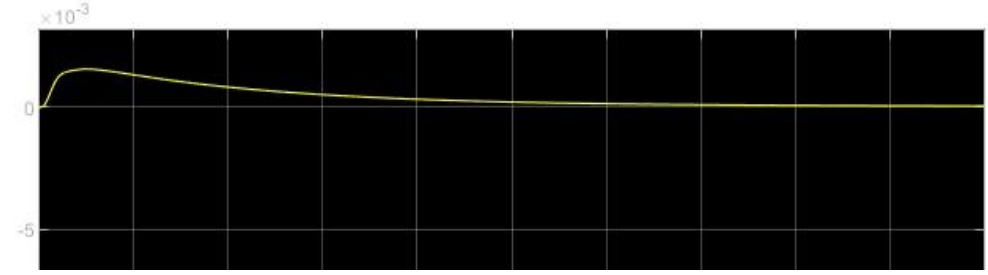




Naturalmente la stima della posizione è relativa alla linearizzazione, perciò i 0.021 metri di equilibrio sono gli 0 metri del regolatore. Come osserviamo, la velocità vera presenta delle oscillazioni (dovute al ripple del PWM) che invece quella stimata non presenta. Volendo visualizzare l'integrale dell'errore, abbiamo:



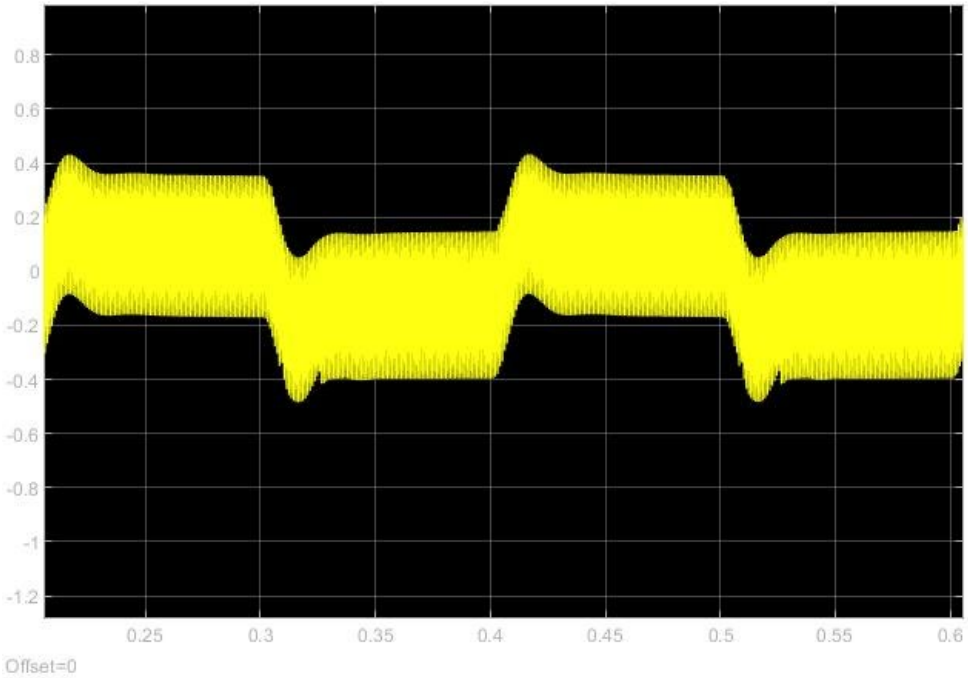
Quest'ultimo si stabilizza a -20e-3, nuovamente a causa del ripple. Infatti, se proviamo ad alzare la frequenza del PWM a 100 MHz (così da ottenere un ripple irrilevante), l'integrale dell'errore ha il seguente andamento:

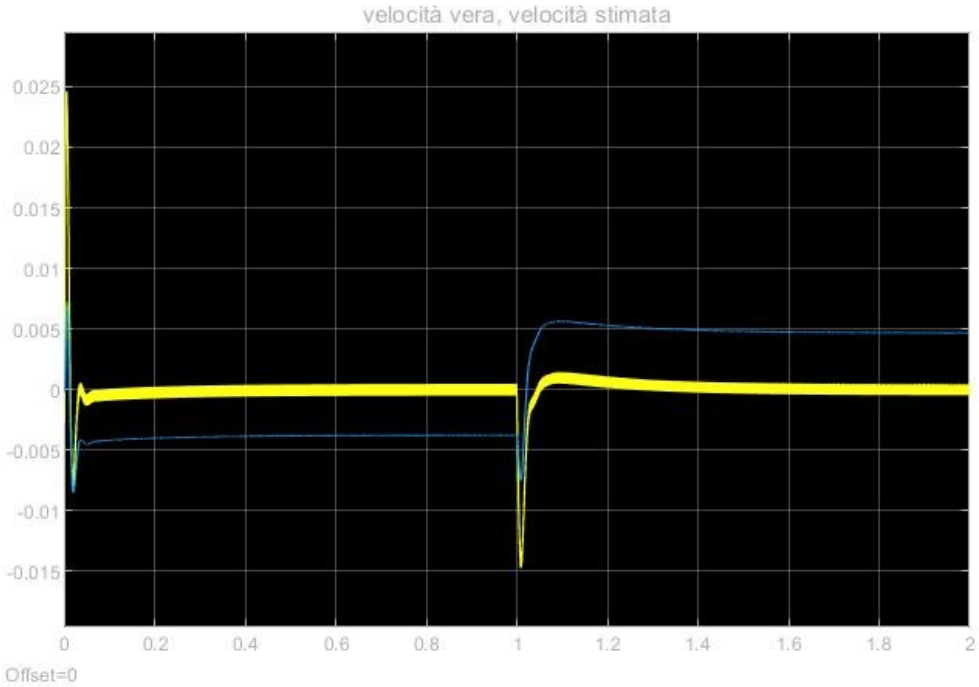


stabilizzandosi infatti a 0.

Disturbi sul comando

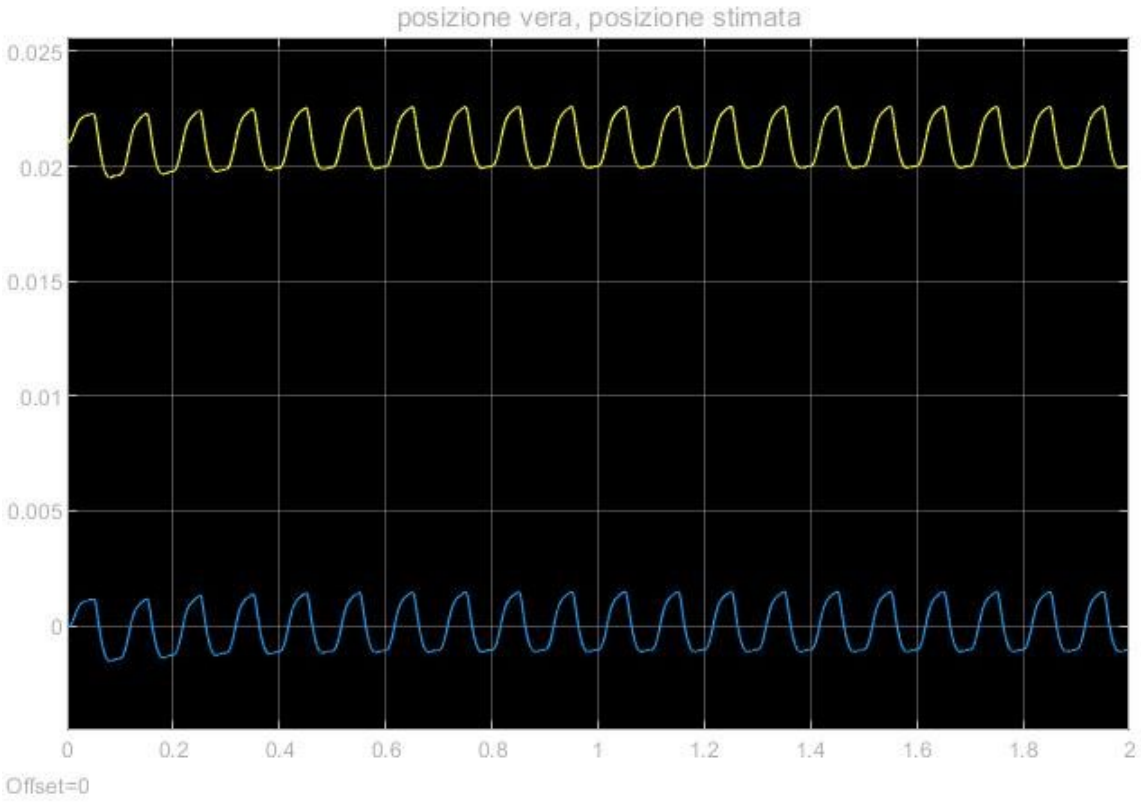
Inserendo un disturbo di 100 mV, 5 Hz sul comando:

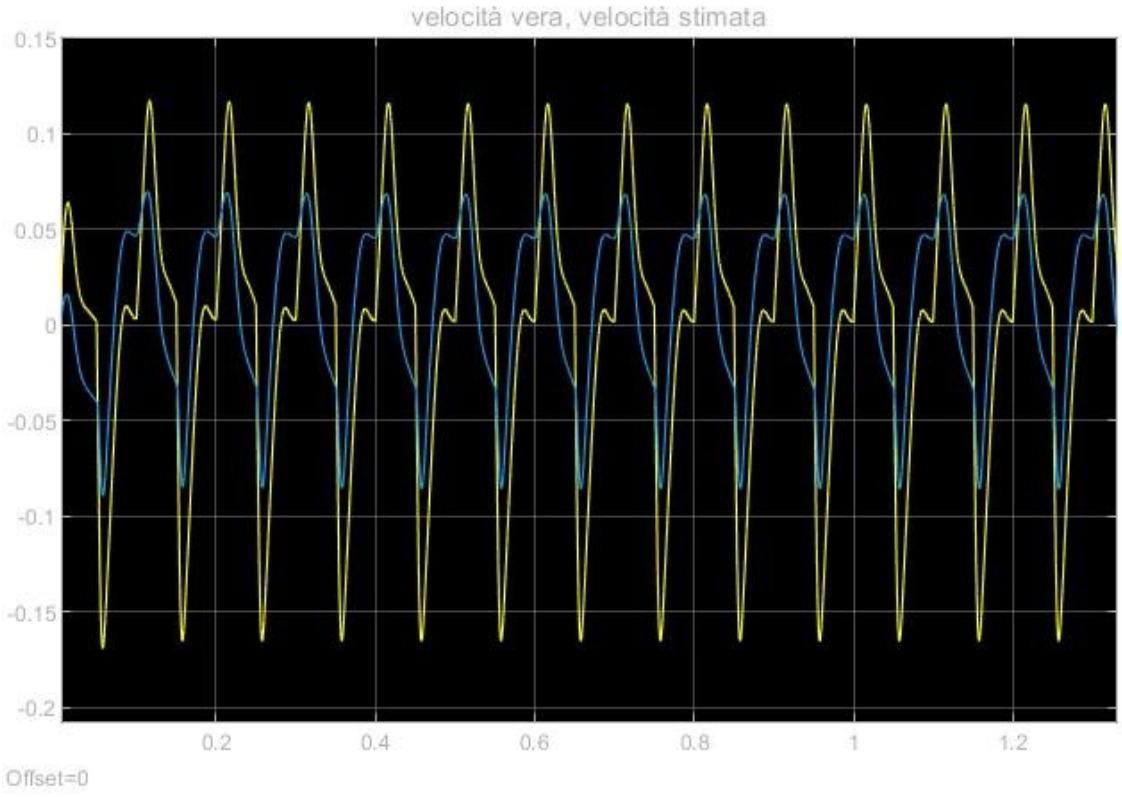




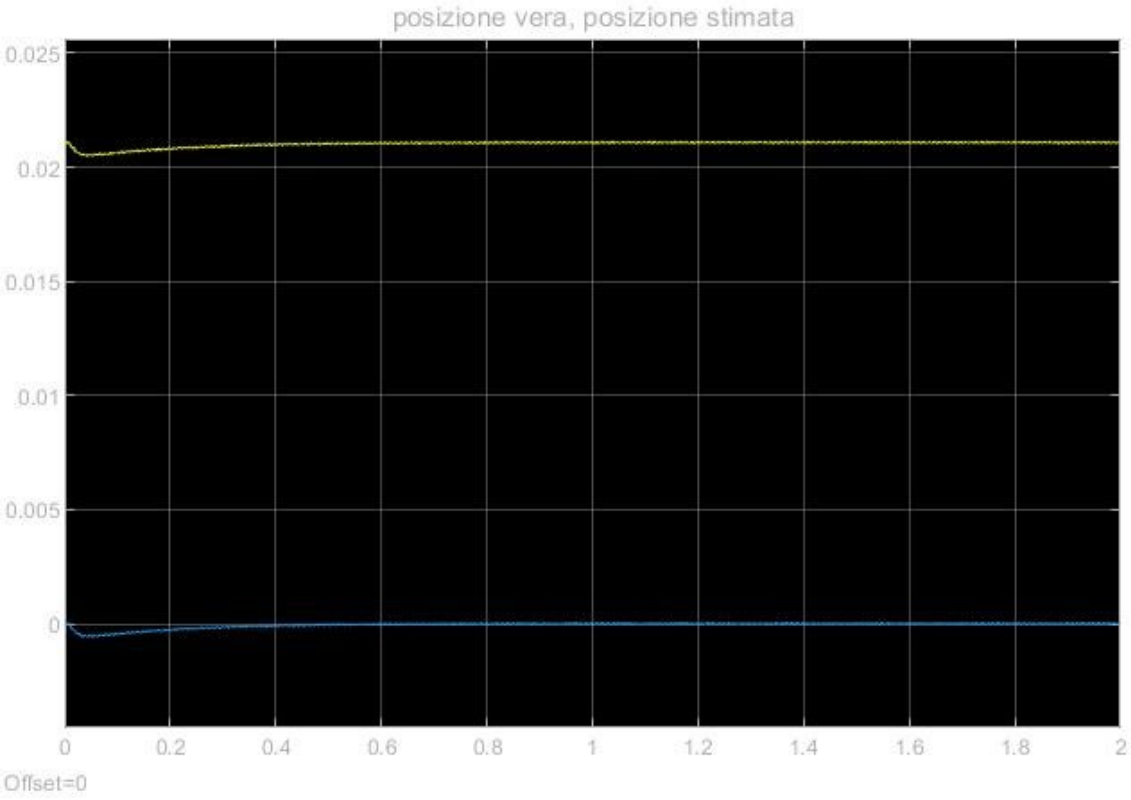
La velocità stimata continua ad essere più “pulita”. Osserviamo che, a differenza di quella reale, non è 0; ciò può sembrar strano ma, se quella stimata fosse 0, il regolatore non avrebbe bisogno di compensare nulla e continuerebbe ad azzerare il comando! Il controllo integrativo è proprio ciò che permette di evitare queste “intermittenze”.

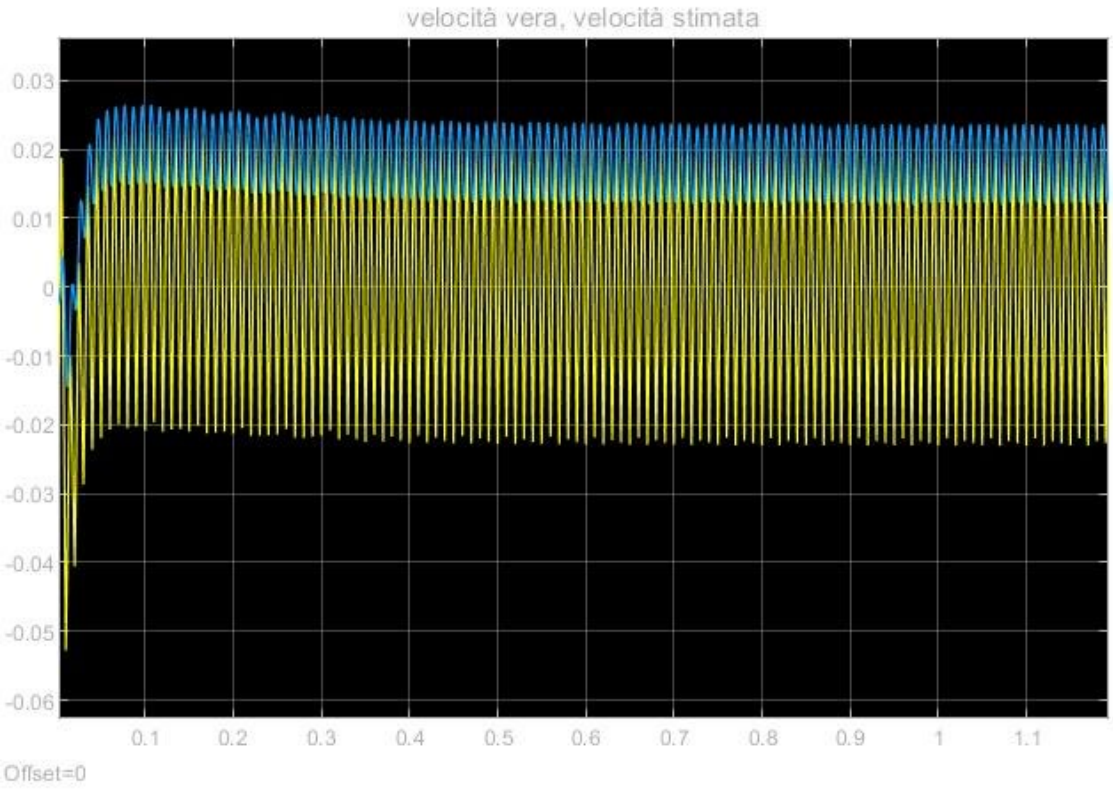
Proviamo a vedere cosa capita con 1 V, 10 Hz:





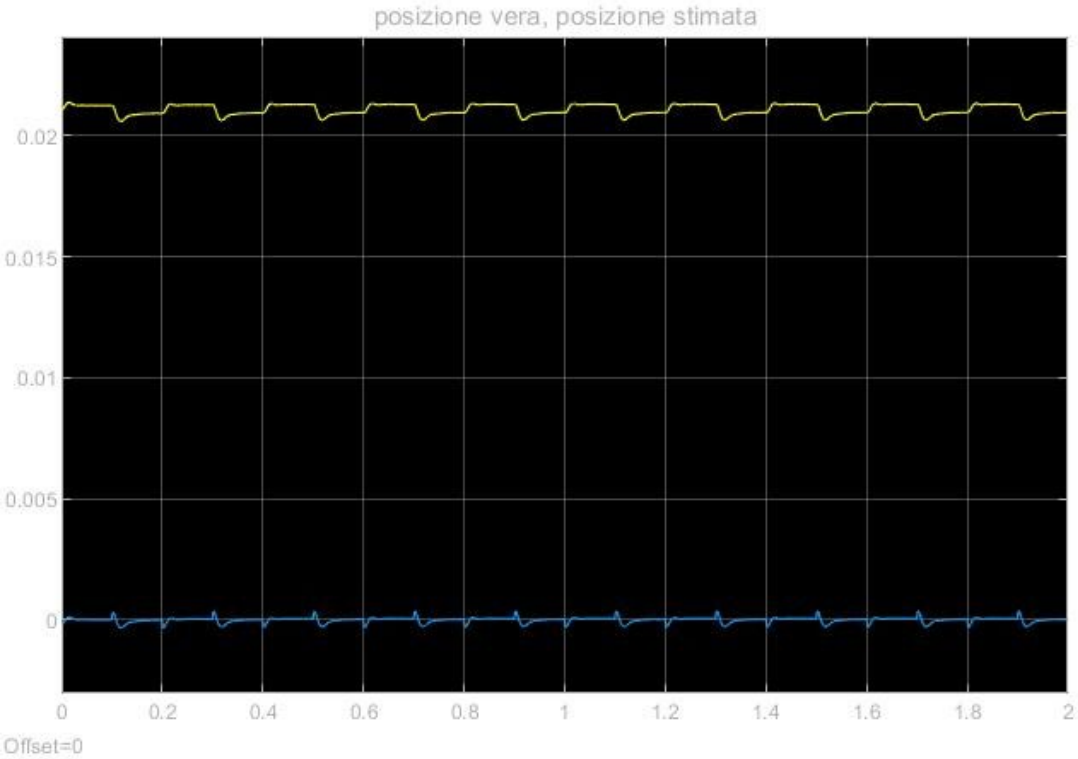
1 V, 100 Hz:

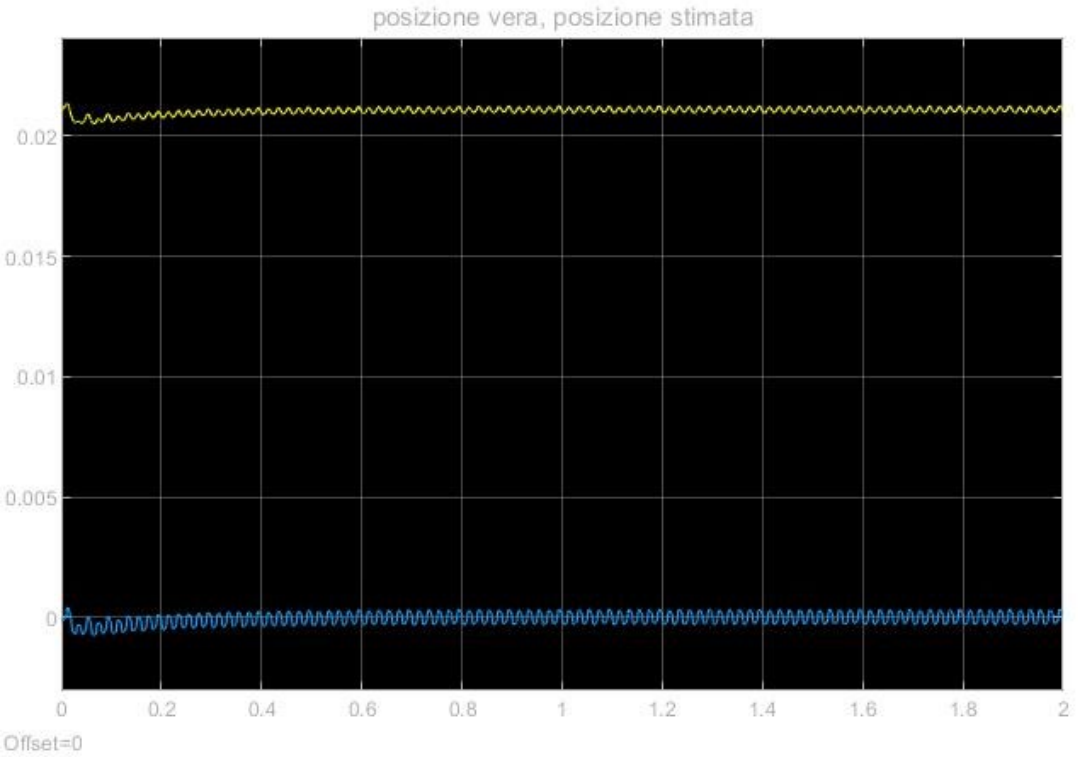

posizione vera, posizione stimata
0.025
0.02
0.015
0.01
0.005
0
0
0.2
0.4
0.6
0.8
1
1.2
1.4
1.6
1.8
2
Offset=0


velocità vera, velocità stimata
0.03
0.02
0.01
0
-0.01
-0.02
-0.03
-0.04
-0.05
-0.06
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
1.1
Offset=0

Disturbi sull'uscita del sistema

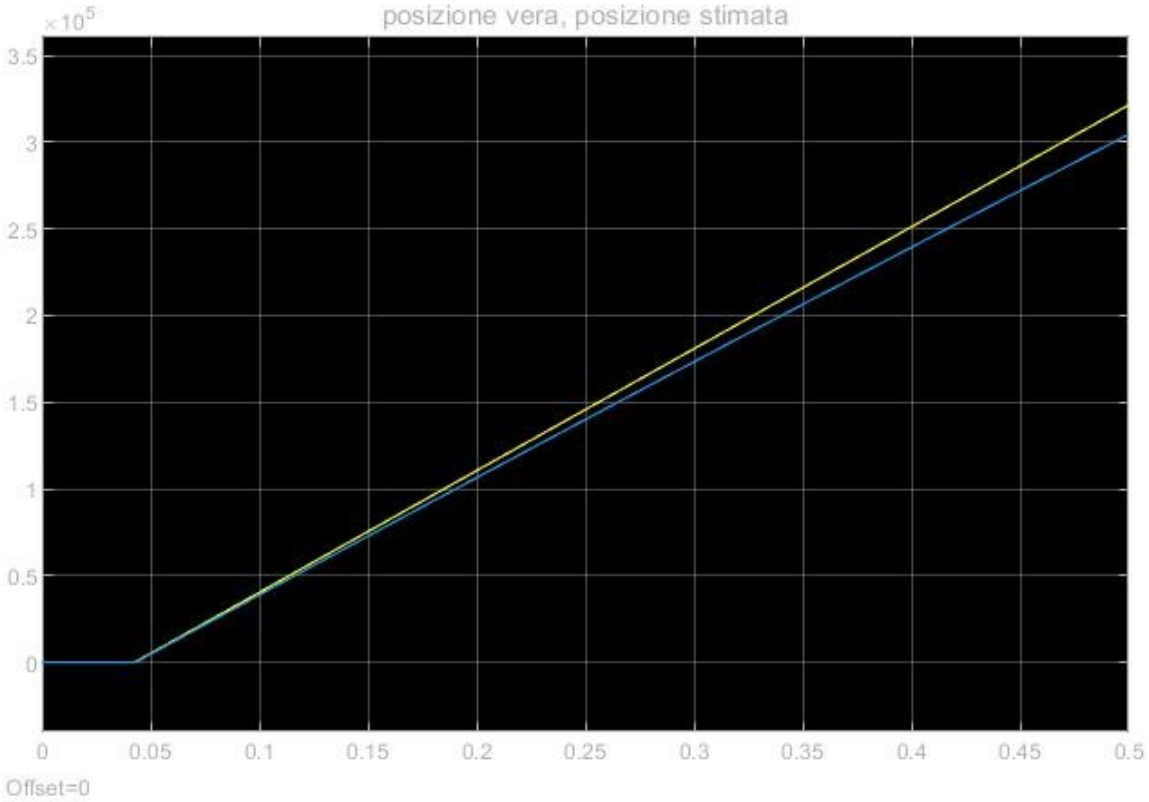
Dando stavolta dei disturbi dopo l'uscita del sistema, subito prima dell'entrata del regolatore, notiamo una sensibilità molto più elevata, come tra l'altro osservato nei capitoli precedenti. Dando ad esempio 0.1 V, 5 Hz:



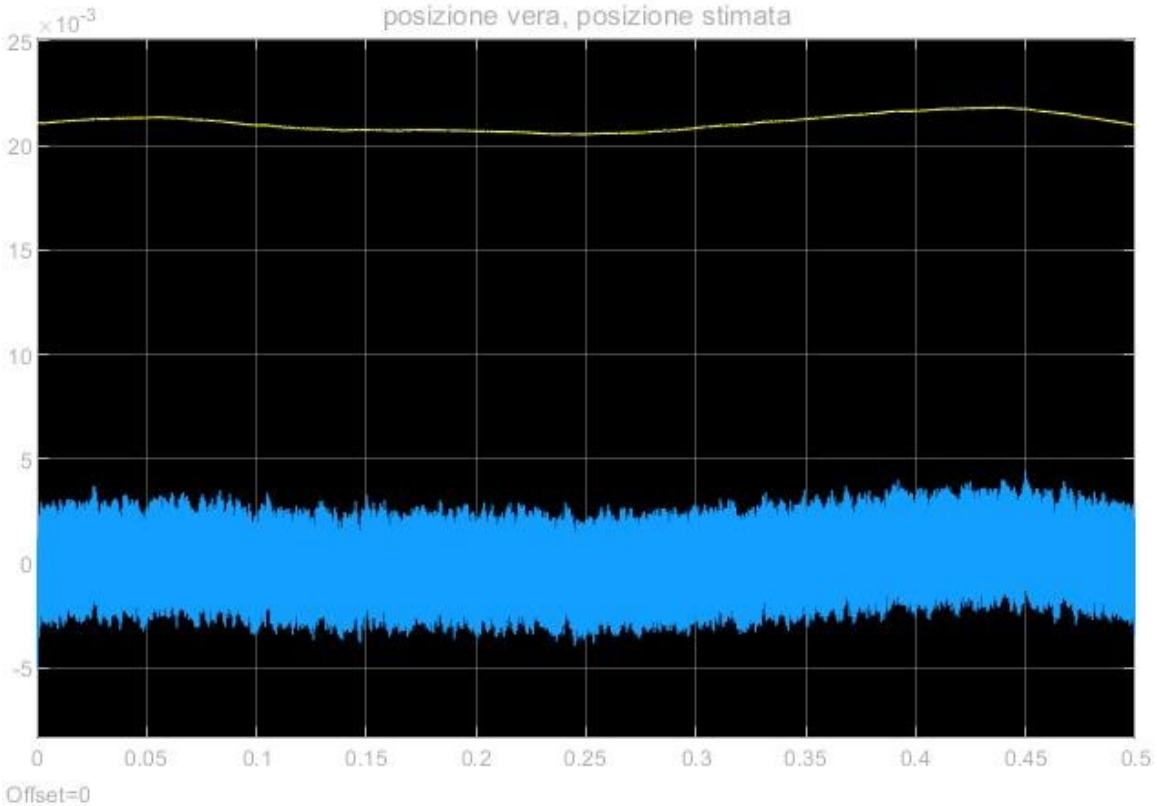


Stavolta anche la stima è disturbata. Inoltre, alzando di poco l'ampiezza, si esce dalla linearità.

Notiamo che, mentre dare 10 V a 100 Hz destabilizza il tutto:



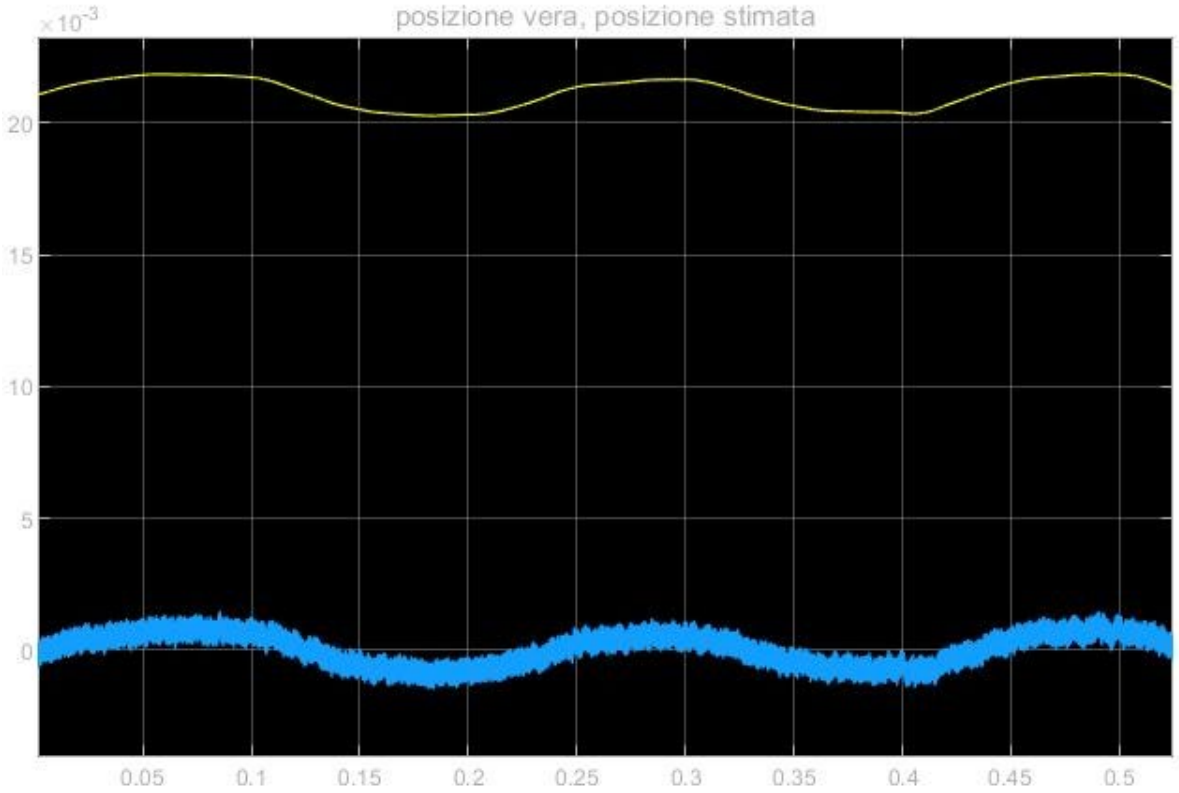
Dare gli stessi 10 V a 10 kHz non lo perturba di molto:



A fronte però di una stima più disturbata.

Ciò è compatibile con le riflessioni fatte nei capitoli precedenti riguardo le funzioni di trasferimento tra i disturbi e le uscite.

Notiamo che abbassando la frequenza dei poli del regolatore (impostando quelli della retroazione statica a [-50, -40, 5]) otteniamo, al costo di un osservazione più “lenta”, una stima più pulita:



Se ne deduce che, come per ogni problema ingegneristico, anche qui ogni cosa ha il suo costo: in questo caso, la miglior velocità ha il costo della peggiore stima, e viceversa.

9. Inseguimento di riferimenti

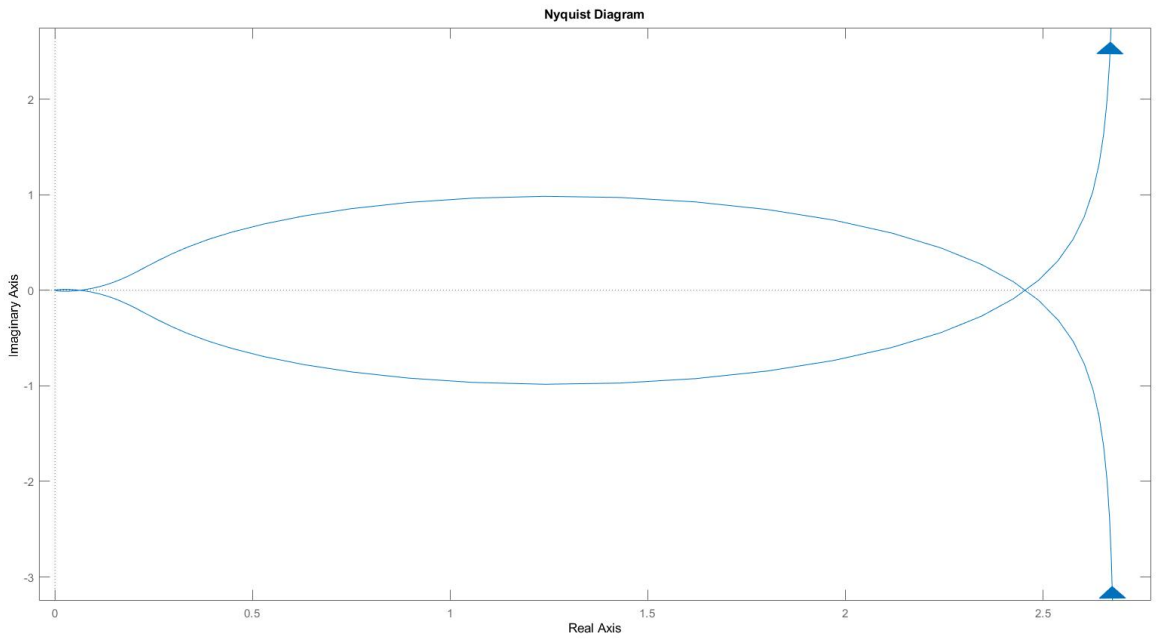
Ispirati dal fatto che quello fino ad ora fatto è stato un controllo applicato agli errori, possiamo dare in ingresso al regolatore non più l'uscita, ma la differenza tra un riferimento e l'uscita: gli stati del regolatore, anche se le matrici resteranno immutate, indicheranno quindi:

- l'errore stimato sulla posizione
- l'errore stimato sulla velocità
- l'integrale dell'errore sull'uscita

Anche se, internamente al regolatore, verranno visti come posizione stimata, velocità stimata ed integrale dell'uscita: questa stima apparentemente errata si rivela in realtà in grado di dare il corretto comando per inseguire riferimenti almeno di grado 1. Infatti, a prescindere della stima, quello che conta è che faccia convergere l'errore: considerando che quest'ultimo sarà stavolta $r - y$, la convergenza implicherà $r = y$. Le t. f. di catena aperta e chiusa sono le seguenti (dove F è il sistema da controllare):

$$G_a(s) = K_c R(s) F(s), \quad W_{ref}(s) = \frac{G_a(s)}{1 + G_a(s)}$$

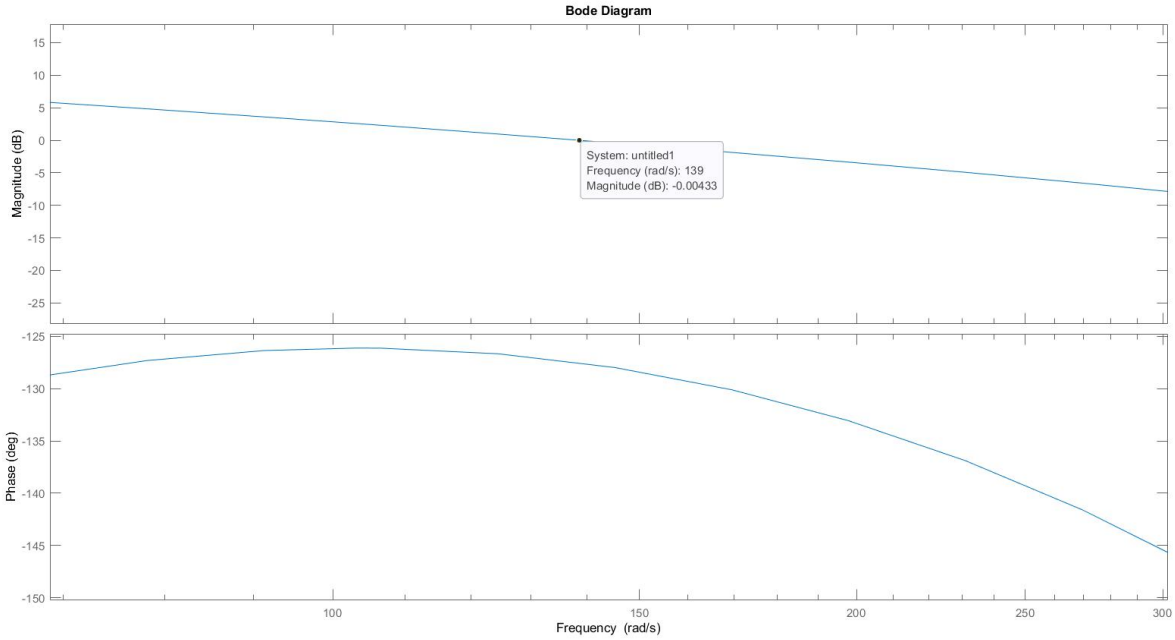
Prima di chiudere la catena, analizziamo il diagramma di Nyquist della catena aperta:



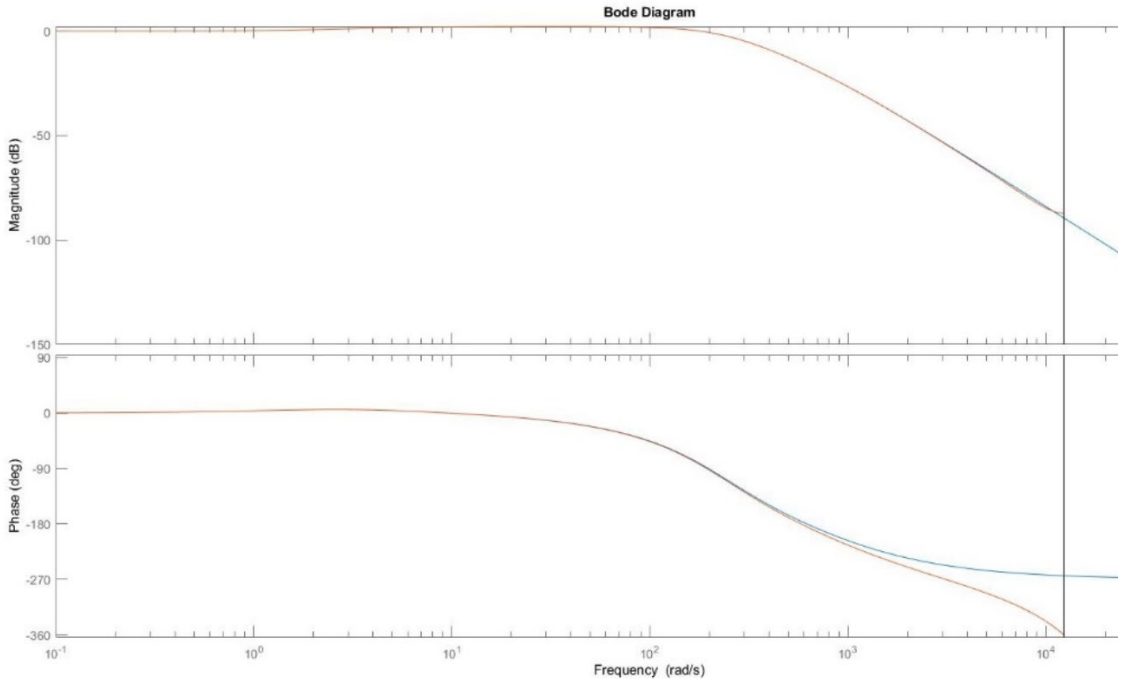
Sapendo che i poli della catena aperta sono 0, -30, 30, -644+115i e -644-115i, occorre posizionarsi nella zona centrale, in cui si ha una sola rotazione antioraria, per bilanciare quel polo instabile.

Otteniamo quindi che Kc dovrà variare da -15.38 a -0.408.

Scegliendo Kc=-2, otteniamo il seguente diagramma di Bode della catena aperta:

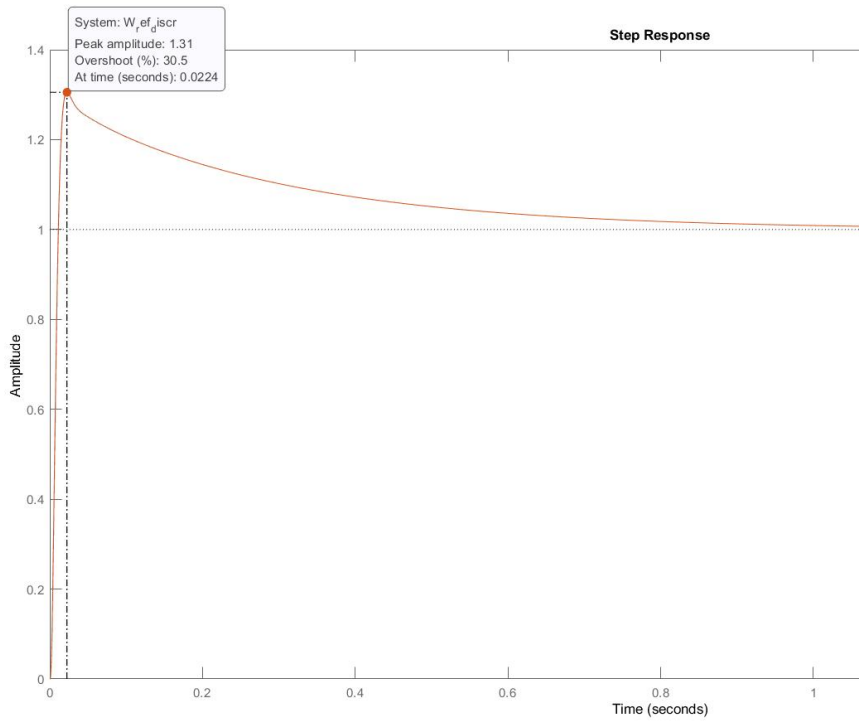


Si nota una frequenza di crossover di circa 139 rad/s con un margine di fase di 50° circa: ci aspettiamo approssimativamente una banda passante in catena chiusa di 220 rad/s e un picco di risonanza di 2 dB. Il diagramma di Bode della catena chiusa è il seguente:



In cui abbiamo sovrapposto al sistema a tempo continuo anche quello campionato.
Abbiamo un picco di risonanza di 2.1 dB a 35.9 rad/s, e una banda passante di 240 rad/s.

Le due risposte al gradino sovrapposte (perfettamente) è la seguente:



Abbiamo una sovraelongazione del 30.5% e un tempo di salita di 10 ms, mentre un tempo di assestamento pari a 79 ms.

Osservare che ha senso parlare di risposta al gradino solo ai fini delle specifiche: in realtà, dare un gradino di 1 V al sistema complessivo, che ricordiamo essere non-lineare, porterebbe il controllore a saturare a causa della brusca variazione. Gradini dell'ordine dei 100 mV non daranno invece questo problema.

Ricapitolando, queste sono le specifiche in catena aperta:

$wc = 139 \frac{rad}{s}$

$m_{phi} = 50°$

Queste quelle in catena chiusa:

$w_b = 240 \frac{rad}{s}$

$M_{r_{db}} = 2.1\ dB$

$s = 0.305$

$t_s = 10\ ms$

$t_{5\%} = 79\ ms$

Per completezza, riportiamo i poli del sistema in catena chiusa continuo e campionato:

CONTINUO:

-3.48e+00

-3.21e+01

-1.86e+02 + 1.42e+02i

-1.86e+02 - 1.42e+02i

-6.44e+02

-8.80e+02

DISCRETO:

9.99e-01

9.92e-01

9.53e-01 + 3.46e-02i

9.53e-01 - 3.46e-02i

8.49e-01

7.99e-01

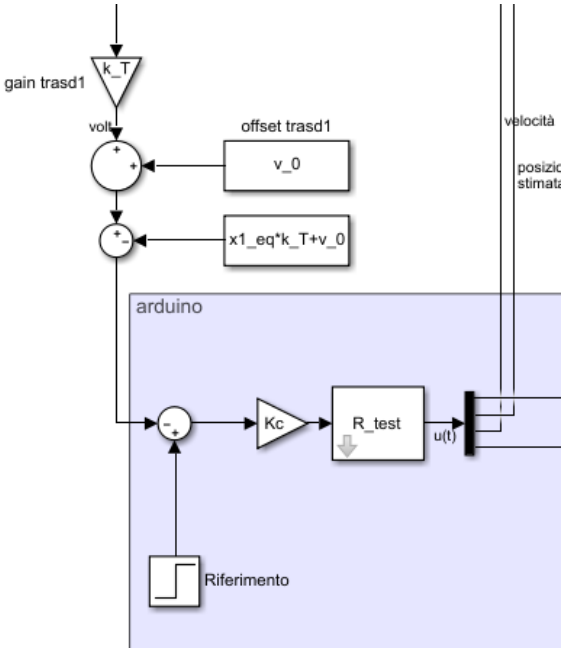
Abbiamo quindi stabilità per entrambi.

La catena aperta ha inoltre un polo in zero, quindi ci aspettiamo un inseguimento perfetto

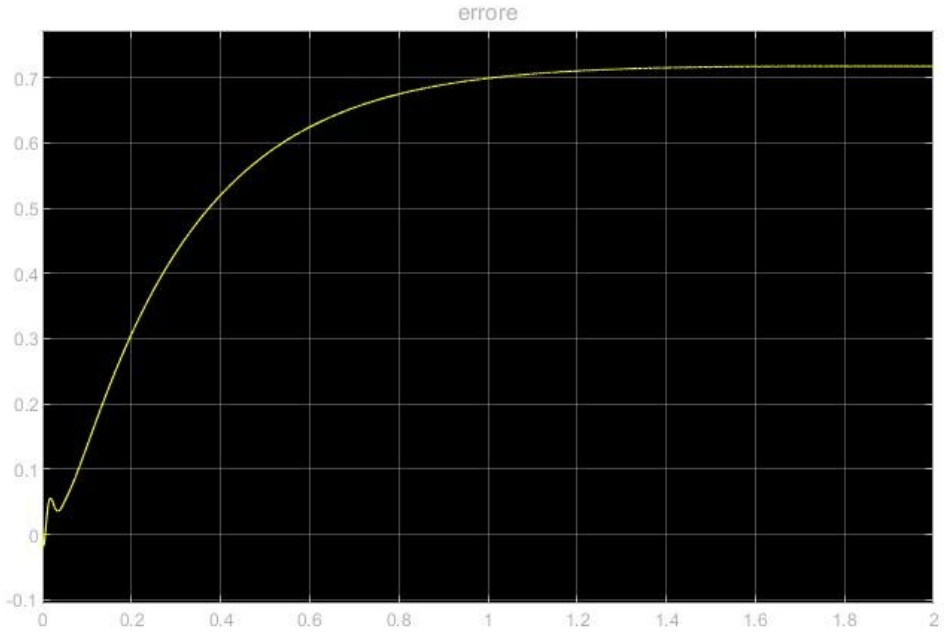
dei riferimenti costanti, con errore finito per riferimenti a rampa, e con errore infinito per riferimenti dal secondo grado in su.

In particolare, per i riferimenti di tipo rampa, l'errore stimato è pari a $\left|\frac{1}{K_f K_c}\right| = 0.065$.

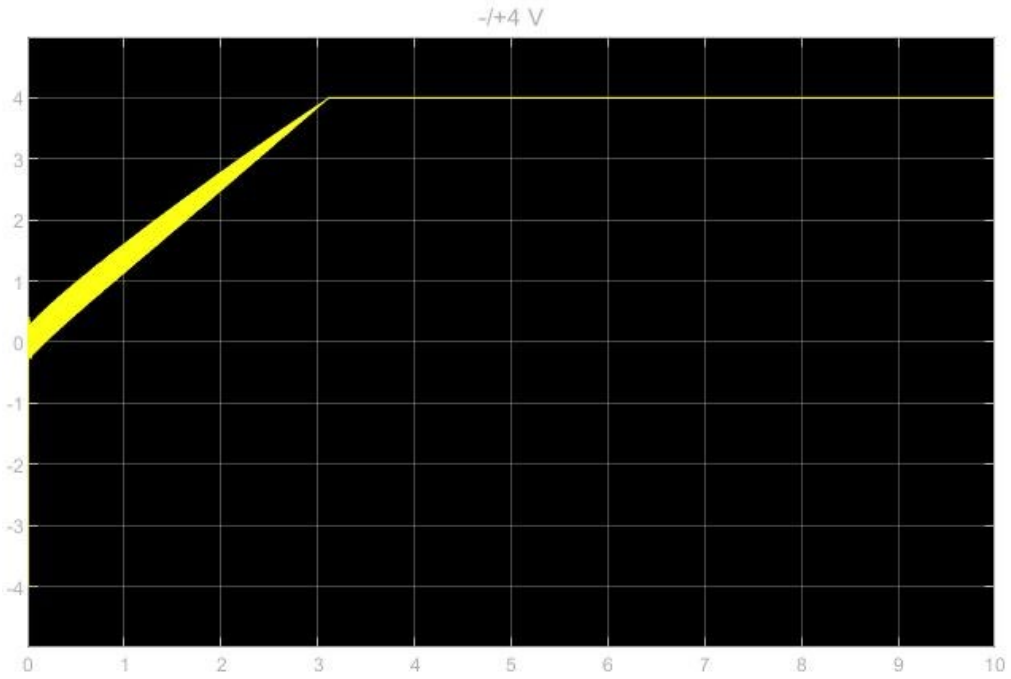
Possiamo a questo punto simulare con Simulink: riportiamo la parte di schema che abbiamo nuovamente modificato:



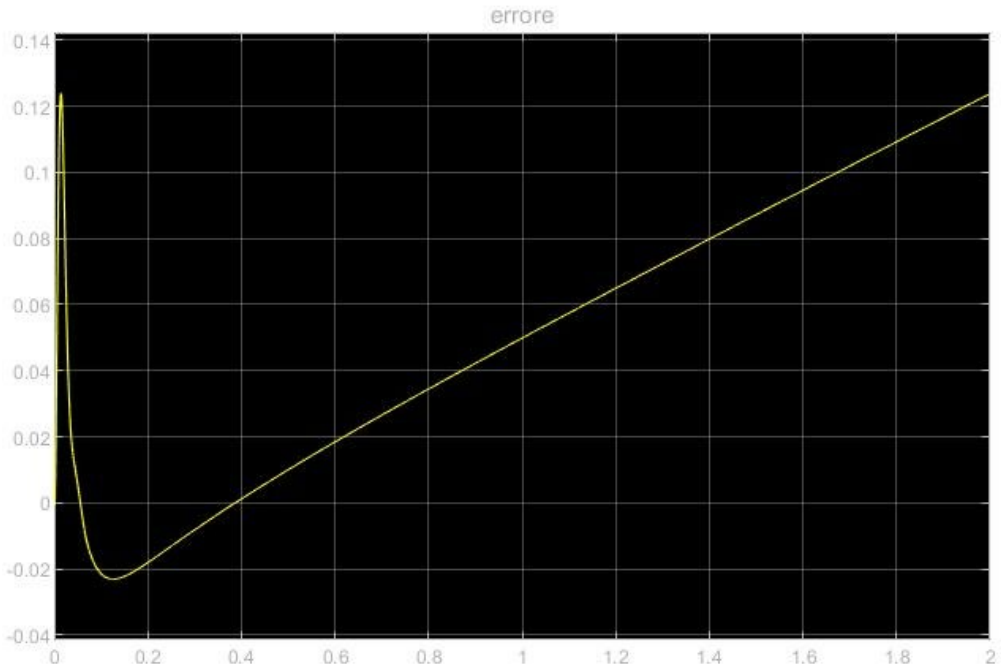
Ribadiamo che, lavorando il regolatore in linearizzazione, il riferimento non indica la vera uscita da inseguire, ma la variazione (rispetto a quella di equilibrio).
Simulando quindi con il regolatore discreto, il PWM e il passa-basso, otteniamo il seguente andamento dell'errore di inseguimento alle rampe:



Il precedente grafico era relativo all'inseguimento di una rampa di pendenza pari a 10 V, perciò l'errore ottenuto è pari a 10 volte quello che si avrebbe inseguendo una rampa unitaria. L'errore è di poco superiore a 0.7 V, il che è compatibile con quanto prima stimato. In realtà occorre tener conto delle non linearità: infatti, dopo 3 secondi l'attività sul comando satura a 4V (che è il valore massimo per cui abbiamo progettato il controllo):

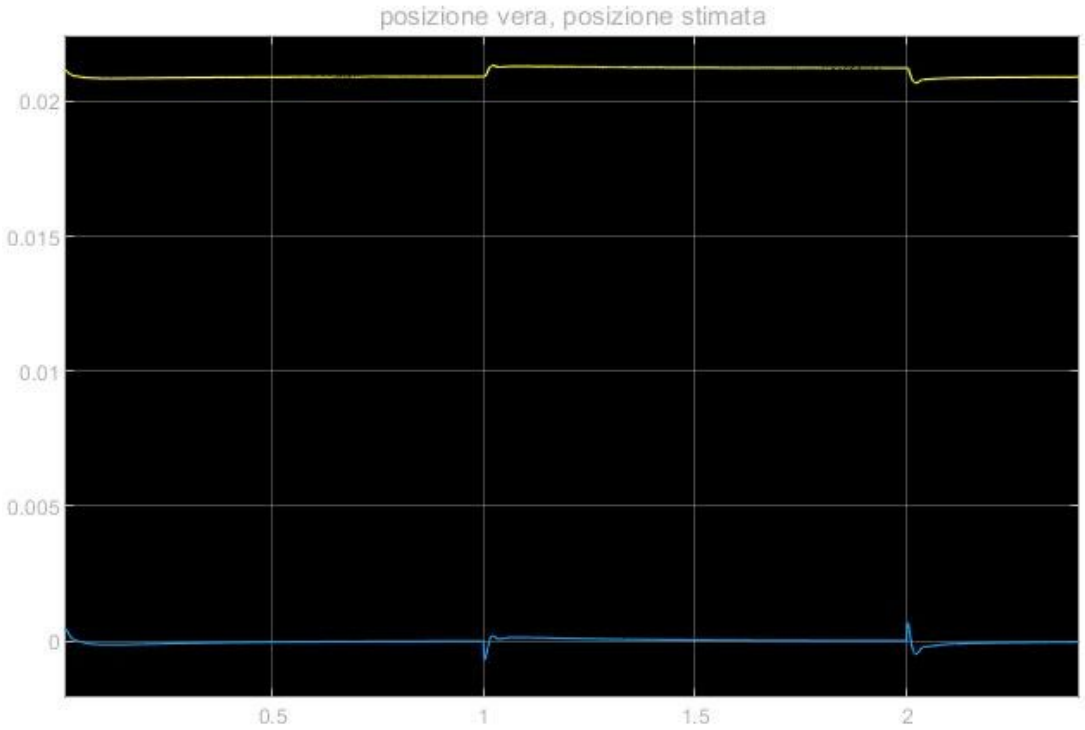


E' comunque possibile abbassare la pendenza della rampa per prolungare la durata dell'inseguimento, ma è chiaro che prima o poi si uscirà dalla linearità.
L'errore per riferimenti parabolici è invece questo:

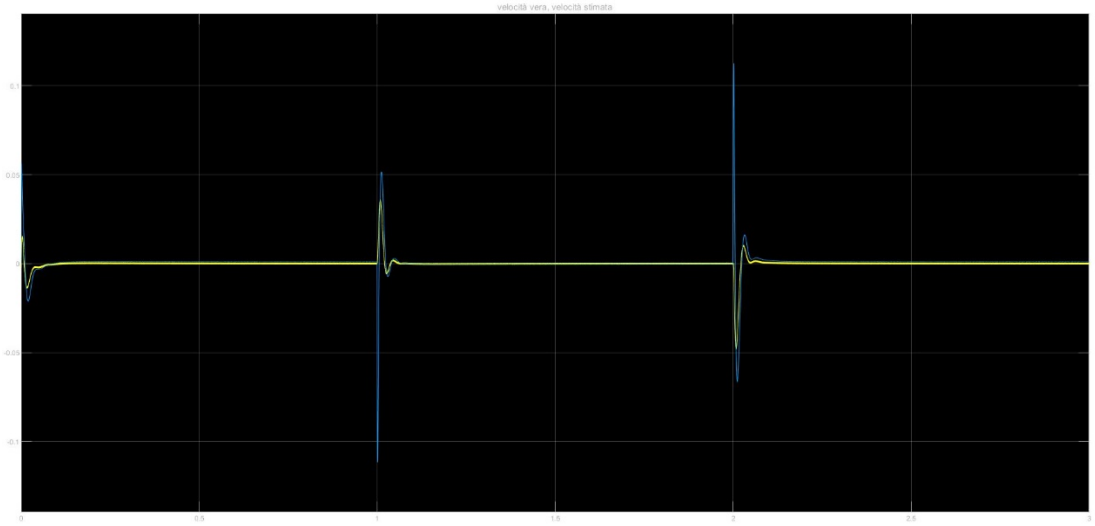


Come previsto, diverge.

E’ istruttivo infine osservare come il regolatore stimi gli stati, dando in ingresso un riferimento a onda quadra, 100 mV, 1 Hz:



Velocità:



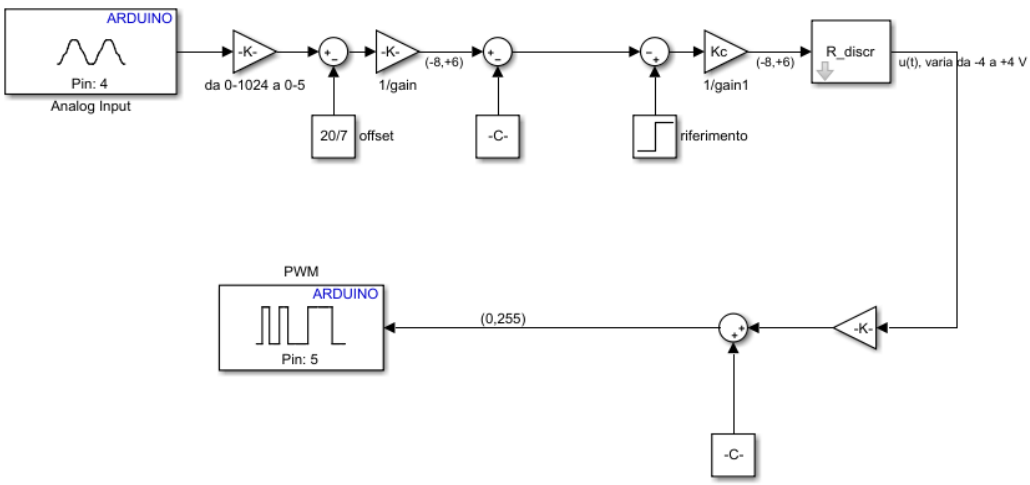
Come ci aspettavamo, anche se la vera posizione non è mai all’equilibrio, per lo stimatore lo è: ricordiamo infatti che il vero stato stimato non riguarda la posizione, ma l’errore tra posizione voluta e posizione ottenuta!

10. Conclusioni

Abbiamo prima testato il controllo statico senza inseguimento dei riferimenti: dopo aver opportunamente collegato i pin di I/O e GND al circuito di interfacciamento discusso nella precedente sezione, abbiamo eseguito il deploy da Simulink e abbiamo sperimentato la buona riuscita del lavoro.
In particolare, dopo aver portato il feedforward al valore nominale e aver sospeso la pallina, quest'ultima è rimasta in equilibrio anche a seguito di perturbazioni di discreta entità, prodotte sia dall'esterno (spingendola con le mani, restando comunque in regione di linearità) sia dall'interno (agendo sulla manopola del feedforward per variare l'ingresso di qualche Volt).
Abbiamo infine sperimentato come il controllo integrativo in particolare facesse sì che, integrando l'errore e portando questo integrale a 0, la posizione tornasse automaticamente a quella nominale.

Successivamente abbiamo implementato la modifica fatta per inseguire i riferimenti, e il comportamento è molto vicino a quello simulato. Il controllo integrativo, quindi, è doppiamente utile: in questo caso, ci ha anche concesso di azzerare l'errore per riferimenti costanti, ottenendo un controllo di discreta qualità sintetizzato nello spazio degli stati, con risultati analoghi a quelli che avremmo avuto usando il loop-shaping in frequenza.

Infine, riportiamo per completezza lo schema Arduino modificato per inseguire i riferimenti:



Se lo si desidera, al posto del blocco "riferimento" (che genera il segnale internamente all'Arduino) si può utilizzare un altro blocco di input per leggere una tensione esterna.

Appendice 1: Misura del ripple reale

Utilizzando il National Instruments VirtualBench, uno strumento che include un oscilloscopio digitale, abbiamo misurato l'entità del ripple causato dal PWM.

Utilizzando il pin out 5 dell'Arduino (PWM a 980 Hz, quello per cui è stato progettato il filtro), abbiamo la seguente cattura dell'attività sul comando:

Voltage

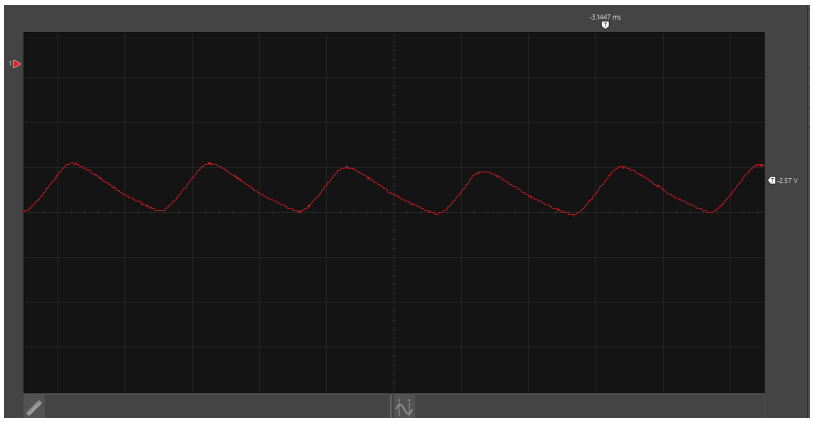
Amplitude	Low	RMS
247 mV	-3.00 V	2.84 V
Peak-to-peak	**Minimum**	**Mean**
535 mV	-3.12 V	-2.84 V
High	**Overshoot**	**Cycle RMS**
-2.75 V	66.5 %	2.81 V
Maximum	**Undershoot**	**Cycle Mean**
-2.59 V	17.0 %	-2.81 V

Time

Period	Pos Pulse Width	Rise Rate
1.0600 ms	639.65 µs	--- V/s
Frequency	**Neg Pulse Width**	**Fall Rate**
943.40 Hz	420.35 µs	736 V/s
Pos Duty Cycle	**Rise Time**	
60.3 %	--- s	
Neg Duty Cycle	**Fall Time**	
39.7 %	267.94 µs	

Notare in particolare come l'ampiezza sia di 247 mV, e la frequenza si aggiri attorno ai 950 Hz (occorre tener conto degli errori di misura del DSO).

Utilizzando invece il pin out 3, che è l'uscita di un PWM a 490 Hz, ci aspettiamo una minore attenuazione, e infatti:



Voltage

Amplitude	Low	RMS
1.11 V	-3.33 V	2.81 V
Peak-to-peak	**Minimum**	**Mean**
1.11 V	-3.33 V	-2.79 V
High	**Overshoot**	**Cycle RMS**
-2.22 V	--- %	2.79 V
Maximum	**Undershoot**	**Cycle Mean**
-2.22 V	--- %	-2.76 V

Time

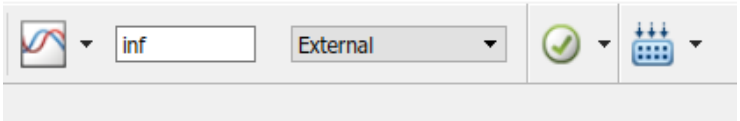
Period	Pos Pulse Width	Rise Rate
2.0600 ms	1.0100 ms	1.76 V/ms
Frequency	**Neg Pulse Width**	**Fall Rate**
485.44 Hz	1.0500 ms	858 V/s
Pos Duty Cycle	**Rise Time**	
49.0 %	506.00 µs	
Neg Duty Cycle	**Fall Time**	
51.0 %	1.0360 ms	

Si osserva un'ampiezza di 1.11 V e una frequenza di 485 Hz, il che è compatibile con quanto simulato.

Appendice 2: Setup di Arduino

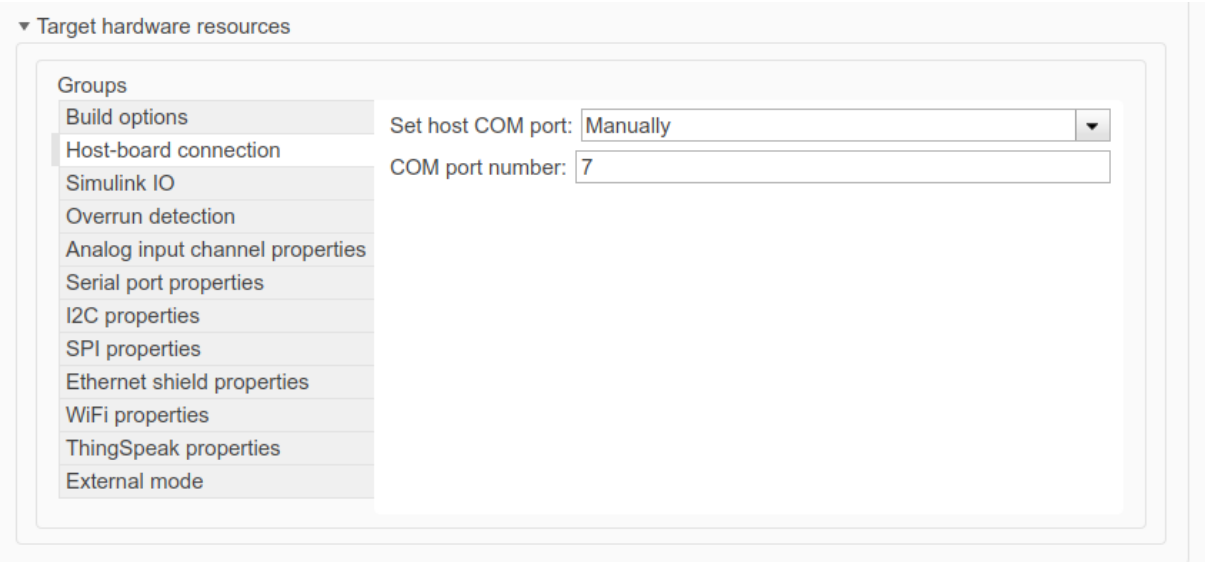
Il package utilizzato per far comunicare Arduino e Simulink è l'Arduino Support Package for Simulink.
Osservare che, anche se la frequenza di campionamento massima è in teoria pari a 9.6 kHz, il rischio di "garbage in" (e quindi "garbage out") è elevato: meglio quindi tenersi a frequenze più basse.

Per visualizzare in tempo reale i valori su Simulink, è necessario utilizzare la modalità "External":



Le prestazioni sono tuttavia molto scarse per frequenze di campionamento elevate: nel nostro caso, ogni millisecondo nella realtà richiedeva un secondo su Simulink.

Andando su Tools->Run on Target Hardware->Options, è possibile selezionare l'Hardware Board (Arduino Uno nel nostro caso). Nelle impostazioni avanzate, è presente in particolare la possibilità di scegliere su quale porta sia presente l'Arduino:



Assicurarsi quindi di scegliere la porta corretta, controllando ad esempio su "Gestione dispositivi" di Windows su quale porta seriale sia rilevato l'Arduino.

Assicurarsi infine di eseguire il tutto a fixed-step e non a variable-step: essendo infatti il sistema campionato, Simulink non è in grado di simulare a step variabili.

L'Arduino Uno utilizzato è quello del Ladispe contrassegnato dall'etichetta "LAD 243".

Bibliografia

[1] Carabelli S., Argondizza A., *Levitatore magnetico: esercitazioni numeriche e sperimentali,* Politecnico di Torino
[2] Calafiore G., *Elementi di Automatica*, CLUT, 2007
[3] Franklin G., Powell J. D., Emami-Naeini A., *Feedback Design of Dynamic Systems,* Pearson, 2018