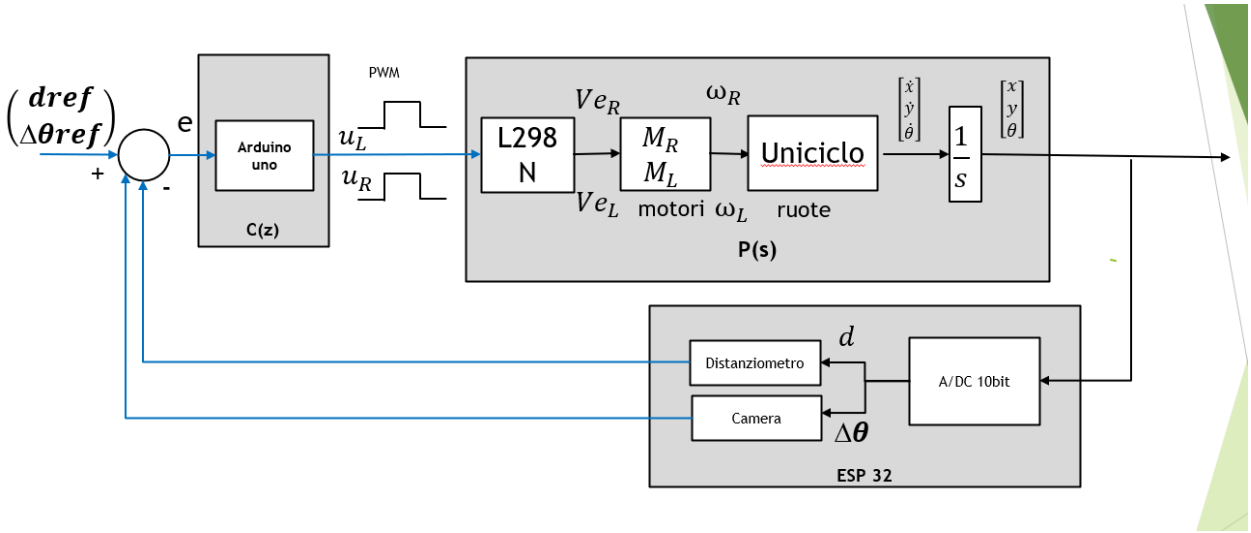
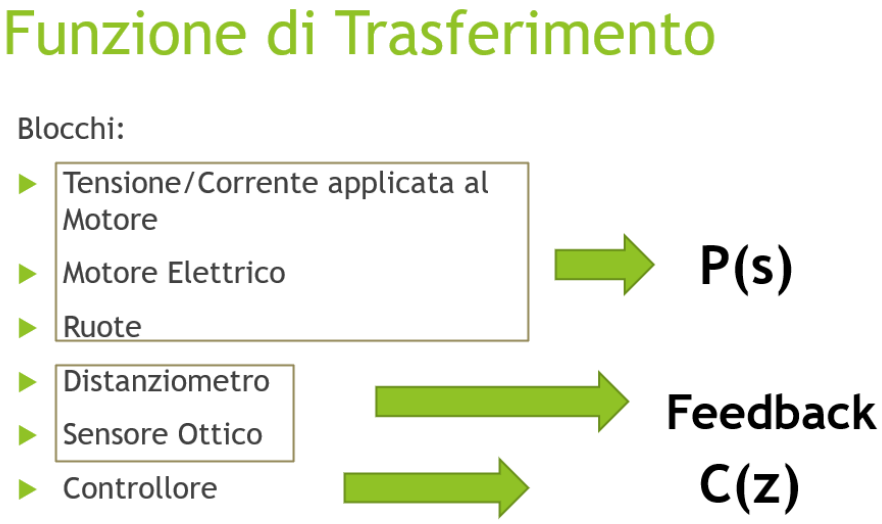


(x,y,theta) è la posa del robot ed è riferita ad un Sistema di Riferimento centrato nel target.
Da questo schema deriveranno due schemi di controllo **indipendenti**, uno per l'angolo theta e uno per la distanza, in modo da risolvere due problemi quasi completamente lineari.
A/DC 10 bit è interno all'ESP e si intende il campionamento delle distanze (d) e della camera (Deltax≈Deltatheta).
Arduino è il controllore discreto, comanda i motori attraverso PWM. (vedi teoria PWM, segnale ad onda quadra, con Frequenza e Ampiezza costante. Il modulo della variabile è proporzionale alla larghezza del livello logico alto, ossia in percentuale "quanto tempo sta al livello logico alto considerando un periodo fissato".Il PWM è direttamente utilizzato dal L298N che comanda le spire del motore (Più il segnale è largo e più L298N fa girare velocemente il motore, a parità di tensione di alimentazione").

Come otteniamo i valori dei vari blocchi?



P(s): Motore Elettrico

[ref: Appunti G.Manca, Presentazione Alex, Appunti carnevale corso DAFN, presentazione E.Donato]

Consideriamo la dinamica meccanica ed elettrica di un motore elettrico. Schematizziamo il circuito di funzionamento:

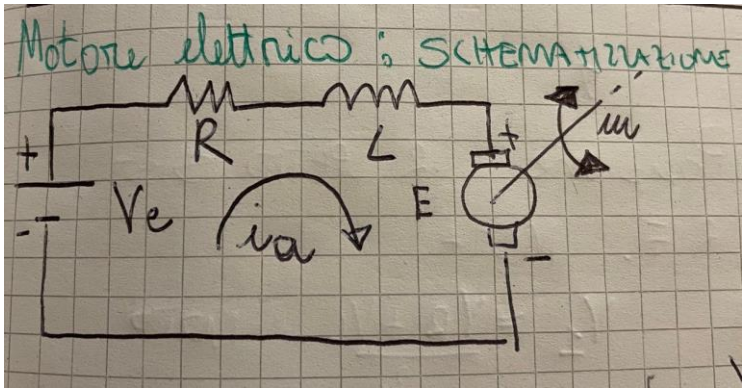


Figura 1: da ridisegnare

Meccanica:

Scriviamo l'equilibrio ai momenti

$J\dot{\omega} = \tau_m - c \cdot \omega + \tau_e$ (1)

τ_m=coppia motrice del motore elettrico

$c \cdot \omega = coppia\ di\ attrito$

τ_e=coppia esterna

J= momento di Inerzia dell'albero del motore + ruote

La coppia applicata è proporzionale alla corrente i_a, la coppia esterna si assume nulla, la (1) diventa:

$J\dot{\omega} = K_m i_a - c \cdot \omega$ (2)

Applichiamo Laplace:

$Js\omega(s) = K_m i_a(s) - c \cdot \omega(s)$ (3)

$\omega(s) = \frac{K_m}{Js+c} i_a$ (4)

È stata trovata la funzione di trasferimento **corrente ->velocità angolare** motore (e ruote).

Circuito Elettrico

Assumiamo comportamento elettrico del motore come resistenza R in serie ad induttore L

$V_e = Ri_a + L\frac{di_a}{dt} - E$ (5)

V_e=Tensione generatore di tensione

E=tensione controelettromotrice dipendente dal flusso di campo magnetico nel motore (**"eddy currents nelle spire interne")**

E è proporzionale alla velocità angolare, quindi $E = K_e\omega$ Applicando Laplace si ottiene:

$V_e(s) = Ri_a(s) + Lsi_a(s) - K_e\omega(s)$ (6)

Dalla meccanica, conosciamo $\omega(s)$:

$V_e(s) = Ri_a(s) + Lsi_a(s) - \frac{K_eK_mi_a(s)}{Js+c}$ (7)

$i_a(s) = \frac{Js+c}{s^2LJ+s(RJ+Lc)-K_eK_m}V_e$ (8)

I poli sono molto veloci, quindi si approssimiamo a:

$i_a(s) \cong \alpha\ V_e(s)$ (9)

Complessivamente:

$\omega(s)=\frac{K_m\alpha}{Js+c}V_e$ (10)

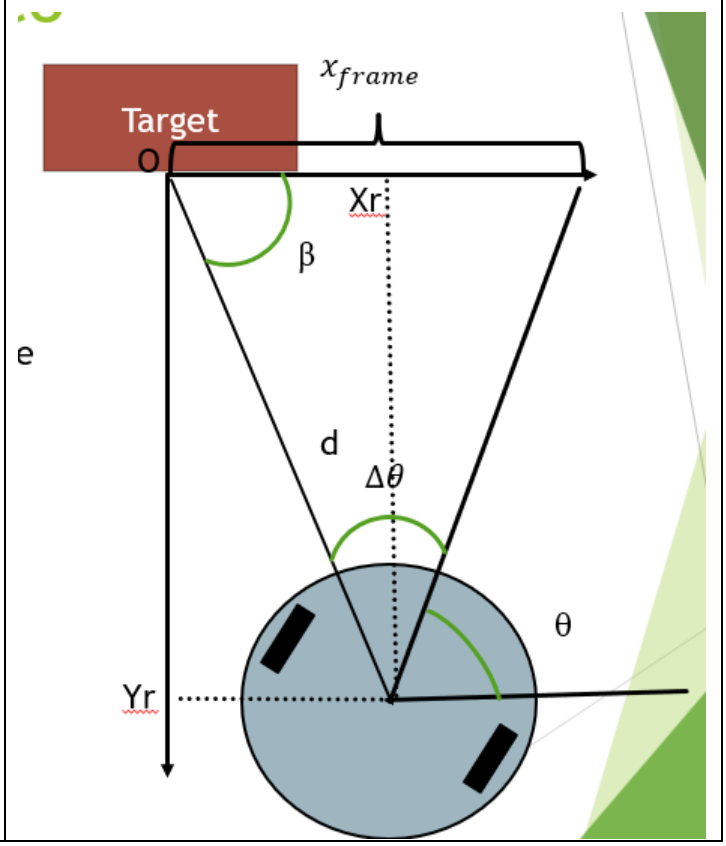
Fdt Ve->omega di rotazione asse motore

P(s): Ruote ->Posa del sistema

- Dalla teoria del controllo dell'uniciclo (F. Martinelli) si ha:
- $\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\theta}\end{bmatrix}=\begin{bmatrix}\frac{(\omega_R+\omega_l)r}{2}\cos(\theta)\\ \frac{(\omega_R+\omega_l)r}{2}\sin(\theta)\\ \frac{(\omega_R-\omega_l)r}{d}\end{bmatrix}$ (11)
- Risulta essere tutto noto, essendo r raggio delle route, d distanza tra le route. Inoltre, tutto è riferito ad un sistema di riferimento inerziale, che nel caso in questione può essere preso come la posizione del target, quindi la posa dell'uniciclo è riferita a tale punto. Istante per istante si può calcolare la posta dell'uniciclo (stimata), integrando (1/s) tale relazione
  - $\boldsymbol{P(s)=\frac{K_m\alpha}{s*(Js+c)}}$

Feedback: sensore ottico

- Sensore Ottico: da feedback su theta
- $\begin{bmatrix} \beta = \text{atan2}(\frac{Yr}{Xr}) \\ xframe = d * sin(\theta - \beta) \end{bmatrix}$
- *Si linearizza la relazione:*
- $\Delta\theta = K_{frame} * x_{frame}$ (12)



Feedback: sensori ultrasuoni

- d=$\|X_r - X_t\|$ Norma della differenza delle posizioni rispetto SR. (X sono vettori)
- Anche lui misura lo stato del Sistema (x e y)
- **È una misura non lineare**, in quanto è sqrt((xt-x)^2+(yt-y)^2).
- **Si semplifica quindi il problema**, si fa allineare il robot al target, la misura di distanza è riferita quindi solo ad un asse, (es. Xt-x).
- La misura di d e di deltatheta vengono mandate in feedback e confrontate con dei valori di riferimento ossia si fa:
- $\begin{bmatrix} \textbf{d-dmin} \\ \Delta\boldsymbol{\theta} - \Delta\boldsymbol{\theta min} \end{bmatrix}$

Filtro anti-Aliasing
(*Scrivere quali rumori sono presenti*)

Vengono automaticamente scartate le misure……..
Notazioni :
Campionatore : ESP
Mantenitore : Arduino

1) Accodamento dei pacchetti del buffer della seriale dovuti alla comunicazione tra ESP(sensore di acquisizione misure) e Arduino (controllore che definisce la dinamica del sistema).
   FACCIAMO CHIAREZZA :
   L'Arduino leggendo pacchetti dalla seriale tramite la policy FIFO (first in first out) ,applica un controllo che determinerà una dinamica di risposta (spostamento del robot). Esso però dovrà permanere nel nuovo vettore di stato per un tempo opportunamente inserito tale da garantire l'applicabilità del teorema di NYQUIST-SHANNON. Nel frattempo, viene applicato il controllo da parte dell'Arduino, l'ESP continuerà a creare campioni e nuove informazioni che verranno accodate nel buffer della seriale (feedback dello stato misurabile), ciò porterà ad un asincronia tra i 2 sistemi di campionamento e controllo tale da creare una difformità fra il nuovo stato del robot appena raggiunto e le informazioni campionate nello stato precedente .

   POSSIBILI SOLUZINI :
   a) Il campionatore dovrà trasmettere meno informazioni al Mantenitore ,in modo tale che tali informazioni possano quanto meno essere calcolate su lo stato ultimo raggiunto tale tempo di trasmissione e regolato dalla variabile updateInterval.

   SOLUZIONE APPLICATA :
   b) Un'altra proposta più efficiente ,potrebbe essere il controllare la saturazione del buffer seriale su cui si appoggia la comunicazione I2C tra ESP e ARDUINO tramite API che restituirà il numero di byte accodati , togliendo i 32 byte ultimi del buffer (che costituirà il pacchetto con i dati più recenti) verranno letti e consumati tutti i byte su la seriale che costituiranno i dati obsoleti .

```
byte_serial  = Wire.available(); // restituisce il numero di byte accodati nel I2C
trash_byte = byte_serial-pkt_size ;
if(trash_byte>0){
  for(int i  = 0 ; i<trash_byte ; i++){ // for che consuma i byte obsoleti su seriale
    Wire.read();
  }
}
```

   NOTA: questo problema e fortemente influenzato dalla dinamica del robot in quanto per poter campionare un nuovo stato il robot dovrà permanere in tale stato per un tempo predeterminato che andrà in funzione della velocità di campionamento del sensore di acquisizione dati (ESP). Quindi il problema si generalizza in un classico problema di QUNTIZZAZIONE che dovrà andare in funzione del SAMPLING del ESP .

2) Disturbo esogeno derivante dall'Environment su cui il robot raggiunge il suo vettore di stato .

   FACCIAMO CHIAREZZA :
   Il disturbo più comune rilevato trattasi delle fughe dell'mattonato .

   SOLUZIONE APPLICATA : come sistema di reiezione a tale disturbo il robot alimenterà i motori a piena potenza per un breve periodo creando una risposta in regime transitorio caratterizzata da un rise time veloce rispetto alla normale dinamica . Tale risposta eviterà la sensibilità a tale disturbo .

3) Disturbo sul campionamento del colore (VISION) .

   FACCIAMO CHIAREZZA :
   Questo disturbo agisce sul nostro sistema quando l'Enviroment e soggetto a condizioni di sottoesposizione o sovraesposizione di luce che tende ad evirare il colore target che si vuole inseguire.
   Inoltre tale problema viene amplificato dal hardware low budget su cui e stato costruito il robot stesso (viene applicata una risoluzione CCIF 320*240).

   SOLUZIONE APPLICATA : Per evitare tale problema si mantiene una soglia di threshold sul riconoscimento del colore da parte dell'algoritmo di campionamento opportunamente calibrata e inoltre vengono adottati colori target basati su colori primari o affini, atti a creare un target singolare da inseguire. Tale soluzione e annoverabile ad un filtro passa banda la cui soglia di treshold viene rappresentata dalla larghezza di banda (invece di proiettare in frequenza tale filtro verrà proiettato su lo spettro colori).

```
#impostazione del colore target in base allo spazio colori HUE-SATURATION-VALUE
#hue -> COLORE
# S  -> SATURAZIONE
# V  -> indica la luminosita del colore

l_h, l_s, l_v = 153, 20,20 # valori minimi
u_h, u_s, u_v = 255, 255, 255 #valori massimi
#la loro differenza indica la soglia di threshold il quale vengono ammessi colori seppur diversi
```

4) Disturbo nella misura delle DISTANZE:

   FACCIAMO CHIAREZZA : il Distanziometro adottato per le misure sfrutta la velocità del suono per computare la distanza fra un oggetto e il robot stesso .
   Di base il concetto su cui si basa e molto semplice viene misurato il tempo di viaggio del suono dal momento di emissione al momento della ricezione dello stesso . Da tale tempo poi verrà estrapolata la distanza effettiva conoscendo la velocità del suono effettiva.
   Il disturbo agente su tale sensore è dovuto al fatto che il suono non torni correttamente al ricevitore per vari motivi , uno fra tutti sono i spigoli di un muro o eventuali oggetti troppo fini .

SOLUZIONE APPLICATA :
la soluzione a questo disturbo si basa sul determinare un valore caratteristico in caso di errore (1.000) . questo valore verrà discriminato in fase di applicazione della legge di controllo eseguita dall'Arduino

```
// calcolo della distanza
digitalWrite(TRIG_PIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIG_PIN, LOW);
ping = pulseIn(ECHO_PIN, HIGH,SONIC_TIMEOUT); //misura la durata dell'impulso dal pin ECHO
if(ping !=0)
  distance_cm = ((ping*VELOCITA_SUONO)/2)/10000;  // calcolo della distanza scalandola in cm
else
  distance_cm = MAX_DISTANCE; // in caso di errore verrà scritto 1000 nella variabile
```

Ritardi:

Consideriamo **ritardo** di 0.5s (Ritardo max circa pari a 2 secondi ma è abbissale).
T di campionamento **Ts**`=1/(29 Hz);`

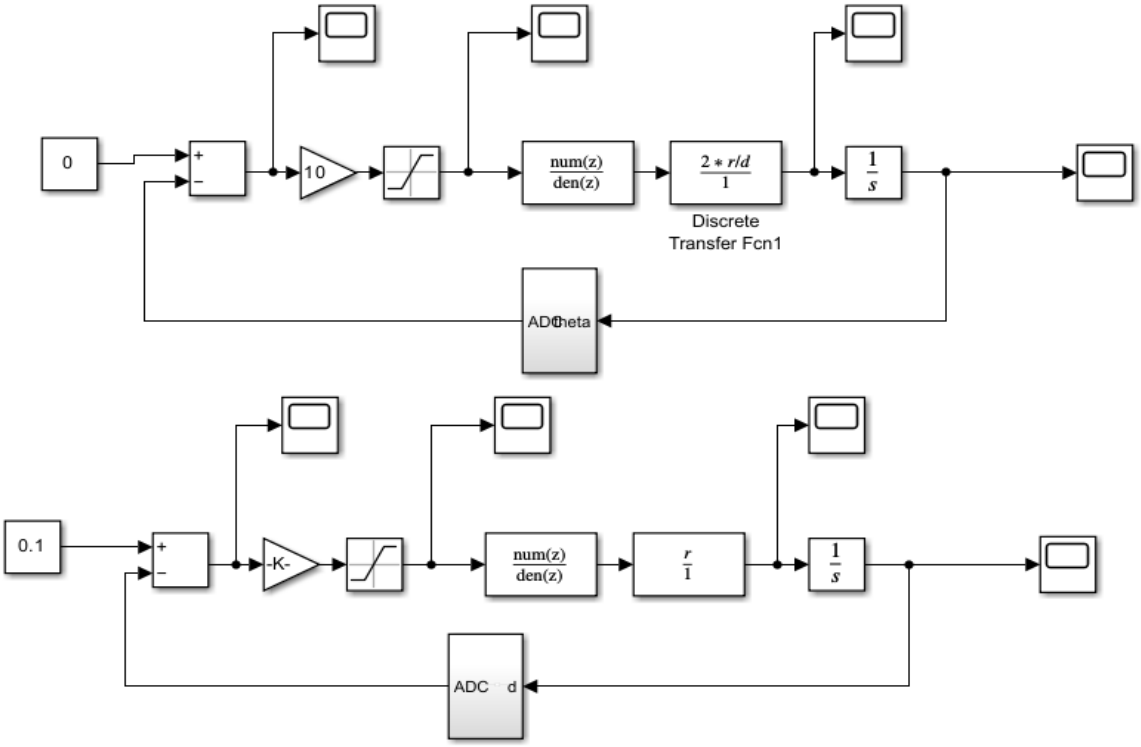
`Il ritardo di campioamento dell ESP e dovuto principalmente alla velocità di elaborazione del campione acquisito tramite VISION . Tale campione viene aquisito dal calcolatore tramite webpage stream creata dal robot. Dopo l'acquisizione immagine da parte del calcolatore esso provvederà a processarla e ad estrapolare le coordinate x e y dell'oggetto target. Verranno immediatamente mandate all'ESP tramite socket UDP. A sua volta L'ESP impacchetterà il dato e lo trasmetterà all'Arduino. Tale procedura e prettamente vincolata alle velocità di trasmissioni dei dati nella rete locale dove sono connessi l'ESP e il calcolatore .Tale Velocità può oscillare da circa 1 secondo (CASO PEGGIORE), fino a qualche centesimo di secondi (CASO MIGLIORE), quindi un campionamento che oscilla da 1HZ fino a circa 29Hz(di picco). Tale oscillazione non deterministica e prettamente dovuta alla saturazione dei buffer di trasmissione, dell' interfaccia di rete del router nonché anche l'interfaccia di rete del campionatore che sappiamo sia alquanto economica. Tale dinamica genera un annoso problema che crea il conseguente disturbo su le misure trattato nel paragrafo precedente (vedasi Problematica 1).`

Sistemi di controllo:

L'uniciclo opera in due fasi:
a.**Ricerca target**
b.**Raggiungimento target**

a. In questa fase le ruote si muovono in verso opposto con stessa velocità angolare. Ciò implica che theta_dot=omega*2*r/d (da eq 11). Integrando (1/s), la telecamera da feedback su Deltatheta
b. In questa fase le ruote si muovono con stesso verso e con stessa velocità angolare. Ciò implica che v1=d_dot=omega*2*r/2=omega*r (da eq 11). Integrando (1/s), il distanziometro da feedback sulla distanza dal target.

Schematizziamo i due sistemi nel seguente modo:

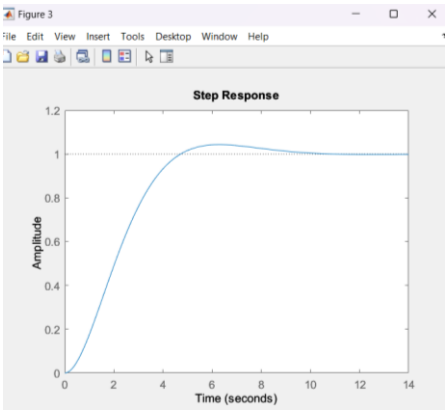


Simulazioni matlab (In progress, scrivere più dettagli)

```
%Parametri Progetto
Km=1; %[N*m/A]
alpha=1;
J=1; %kg*m^2
c=0.5; %N*m*s
r=0.04; %[m]
d=0.2; %[m]
omegaR=0;
omegaL=0;
dmin=0.1; %distanza rif
thetamin=0.2; %angolo rif
Kv1=1;
Kv2=1;
Ts=1/29; %T di campionamento
yold=0;
```

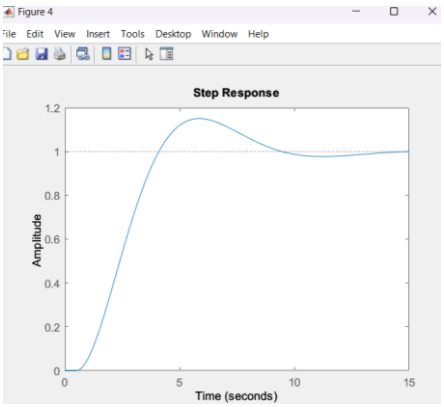
```
%% Simulazione a tempo continuo
%Specifiche: Errore nullo per riferimento costante.
s=tf('s');
P1=Km*alpha/(s*(J*s+c));
C1=0.5;
L1=minreal(C1*P1);
figure(1)
rlocus(L1);
%I poli del sistema a ciclo chiuso si trovano nel semipiano sinistro
figure(2)
[Gm,Pm,Wcg,Wcp]=margin(L1);
margin(L1);
ritardomax=Pm*(pi/180)/Wcp
Wr1=minreal(L1/(1+L1));
figure(3);
step(Wr1);

```

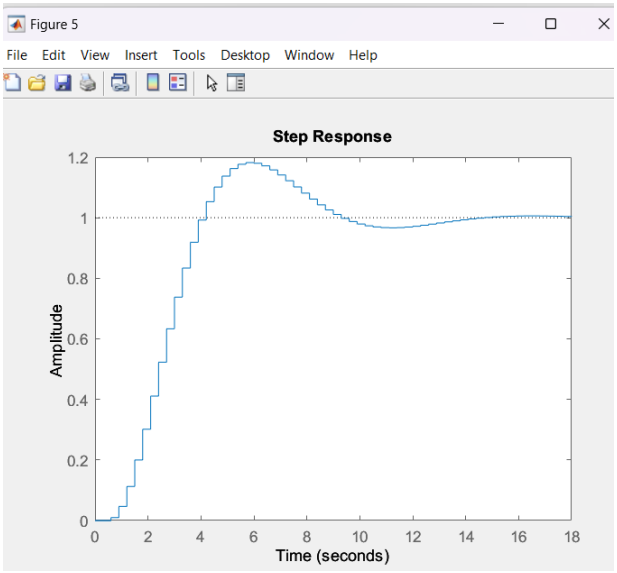



```

%% Introduciamo ritardo
rit=exp(-0.5*s)
Lr=minreal(rit*L1);
Wrr=minreal(Lr/(1+Lr));
figure(3);
step(Wrr);
```




```
%% ** DA RIVEDERE***
% Utilizziamo ora trasformata Bilineare di Tustin per il design del controllore C(z)
%trasformo P(s)->P(z)
Pz1=c2d(Lr,0.3,'tustin');
%Una volta entrati nel dominio w utilizziamo le tecniche di modellazione a
%t continuo. Infine, bisogna fare l'antitrasformate e verificarne in
%funzionamento.
Wz1=minreal(Pz1/(1+Pz1));
step(Wz1)
%% Infine simuliamo su simulink il controllore trovato
```



Simulink:

I sistemi partono da condizioni diverse da zero, ossia la posizione dell'uniclo non corrisponde a quella del target. Tende poi asintoticamente a d_ref e deltatheta nullo.

Figura 2 delta theta

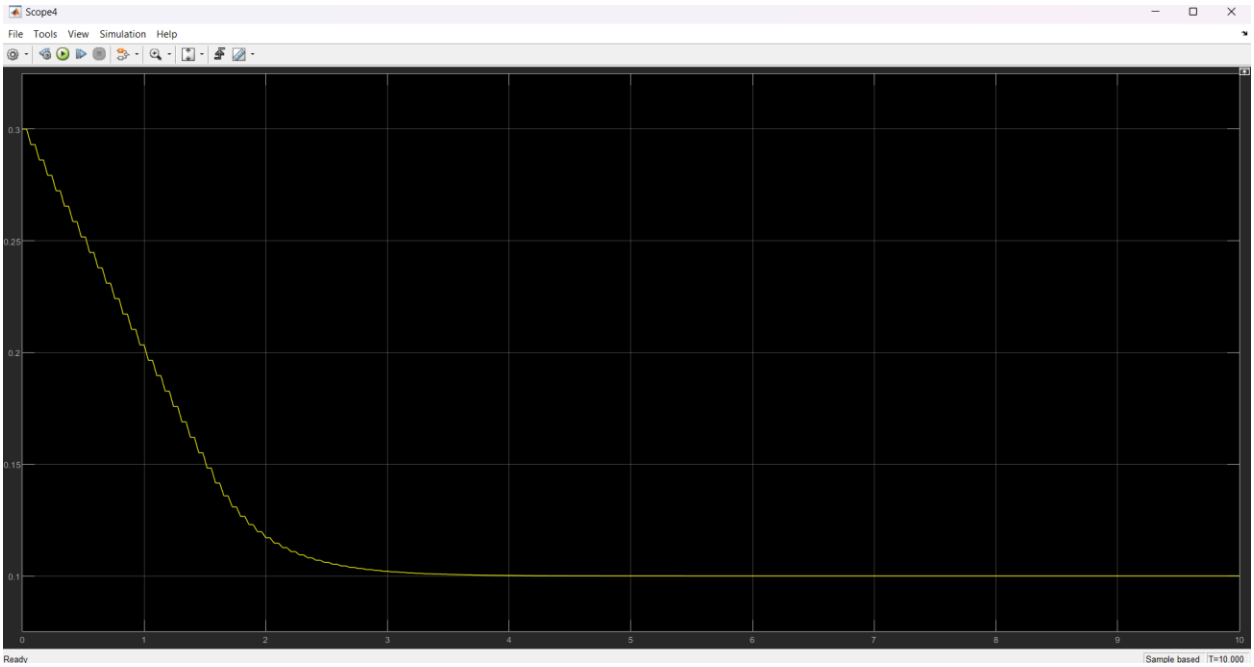


Figura 3 d_ref