UNIVERSITA' DEGLI STUDI DI PADOVA
FACOLTA' DI INGEGNERIA

Corso di laurea in
INGEGNERIA DELL'INFORMAZIONE

Tesi di laurea triennale dal titolo:

Progettazione di un sistema di "cruise control" con implementazione Matlab/Simulink

Relatore:
Prof. Alessandro Beghi

Laureando:
Marco Giordani

Anno accademico: 2012/2013

Indice

INTRODUZIONE

Il cruise control è un sistema elettronico di controllo e regolazione automatica della velocità per autovetture.
È una caratteristica di comodità, che si apprezza soprattutto nei lunghi viaggi. Agendo semplicemente su una leva posta sul volante dell'automezzo, il conducente può impostare una "velocità di crociera", che sarà quindi mantenuta costante. Il guidatore può aumentare la velocità semplicemente premendo il pedale dell'acceleratore, ad esempio per la necessità di sorpassare un altro veicolo ma, una volta rilasciato, l'autovettura ripristinerà il valore di velocità preimpostato. Premendo invece il pedale del freno, il cruise control interrompe la sua funzione e dovrà essere rimesso in funzione agendo manualmente.
Il cruise control è utile non solo per comodità del guidatore, che non deve preoccuparsi di modificare di continuo la velocità del proprio veicolo, ma soprattutto per avere un risparmio economico nel consumo di carburante, dato che vengono evitate delle continue modifiche della velocità che si avrebbero invece con una guida manuale.

Tale elaborato ha lo scopo di studiare e quindi simulare il funzionamento del cruise control, descrivendone il sistema e derivando un modello matematico come sistema lineare.
In seguito si passerà all'analisi dinamica del sistema in catena aperta (risposte indiciali e in frequenza), progettando un controllore mediante diversi approcci (PID, luogo delle radici, reti correttrici) e esaminando infine l'analisi dinamica in catena chiusa (reiezione dei disturbi e verifica del soddisfacimento delle specifiche di controllo).
Alla fine di questi passaggi saranno fatte delle considerazioni conclusive al fine di commentare ed analizzare i risultati ottenuti.

Per rendere più chiare le fasi di questo elaborato, ci si servirà dell'ausilio di *Matlab* e *Simulink*.

CAPITOLO 1
Modellizzazione del sistema

1.1 Configurazione fisica

Abbiamo già avuto modo di osservare nel paragrafo introduttivo che la funzione principale del cruise control è di realizzare un sistema di controllo in retroazione per mantenere la velocità di un mezzo costante, nonostante disturbi esterni che possono verificarsi, come ad esempio cambiamenti del vento o della pendenza della strada. Ciò è realizzato misurando la velocità del veicolo, confrontandola con quella desiderata e regolando automaticamente l'accelerazione mediante una legge di controllo.

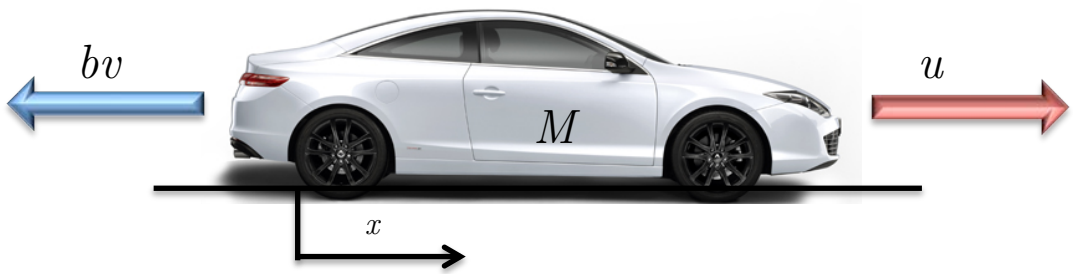


Figura 1.1: Modellino grafico del sistema

Nella figura 1.1 è raffigurato il modellino grafico del sistema per un'autovettura.
Il movimento del veicolo di massa M è originato da una forza di comando u, supponendo di essere in grado di controllare direttamente questa forza e trascurando le singole componenti che la generano. Le forze resistive (bv), dovute alla resistenza al rotolamento e a quella del vento, sono assunte variare linearmente con la velocità del veicolo, v, e agiscono in direzione opposta moto del veicolo stesso.

Si sta quindi considerando un modello dinamico elementare, che è definito dalle seguenti leggi fisiche di velocità e accelerazione:

$$\begin{cases} v = \dot{x} \\ a = \dot{v} = \ddot{x} \end{cases}$$

1.2 Equazioni del modello

Con questi presupposti, è possibile trovare un modello differenziale del primo ordine.

Basta infatti sommare le forze che agiscono in direzione x (che nel modello semplificativo sono solo 2) ed applicare la seconda legge di Newton $(F = ma)$ che mette in relazione forza, massa ed accelerazione.

$$F = u - bv \rightarrow u = M\dot{v} + bv \tag{1.1}$$

Poiché siamo interessati a controllare la velocità del veicolo, l'equazione dell'uscita è scelta come segue:

$$y = v$$

I sistemi del primo ordine, come quello in questione, hanno soltanto un singolo modo di accumulare energia, in questo caso l'energia cinetica della vettura, ed è necessaria quindi una sola variabile di stato: la velocità.

1.2.1 Modello in spazio di stato

In generale, è definito un sistema di equazioni per rappresentare modelli di questo tipo:

$$\begin{cases} \dot{x}(t) = Fx(t) + Gu(t) \\ y(t) = Hx(t) + Ju(t) \end{cases} \tag{1.2}$$

dove F è una matrice quadrata di dimensione $n \times n$ (dove n rappresenta il numero delle variabili di stato), G è una matrice di dimensione $n \times m$ (dove m rappresenta il numero di ingressi), H è una matrice $p \times n$ (dove p rappresenta il numero di uscite) e J è una matrice $p \times m$ (che nel nostro caso è nulla).
Sfruttando questa raffigurazione generale, la rappresentazione in spazio di stato del nostro modello è quindi:

$$\begin{cases} \dot{x} = [\dot{v}] = \left[\frac{-b}{M}\right][v] + \left[\frac{1}{M}\right][u] \\ y = [1][v] \end{cases} \tag{1.3}$$

1.2.2 Parametri fisici del sistema

Per questo esempio e per le simulazioni che saranno condotte, è necessario introdurre dei valori specifici per i parametri incogniti che compaiono nel sistema; tali valori sono riportati nella tabella sottostante:

Descrizione	Variabile	Valore della variabile
Massa del veicolo	M	1000 [kg]
Coefficiente complessivo di attrito	b	50 [Ns/m]
Forza impressa al veicolo	u	500 [N]
Velocità di riferimento	r	10 [m/s]

1.2.3 Implementazione del modello in spazio di stato in ambiente *Matlab*

E' necessario realizzare il modello in spazio di stato, in precedenza descritto, in ambiente *Matlab*. Dapprima devono essere definiti i valori per le incognite, con i seguenti comandi:

```
M=1000;     %Massa del veicolo
b=50;       %Coefficiente complessivo di attrito
u=500;      %Forza impressa al veicolo
F=-b/M;      %Coefficiente dello spazio di stato
G=1/M;       %Coefficiente dello spazio di stato
H=1;         %Coefficiente dello spazio di stato
J=0;         %Coefficiente dello spazio di stato
```

In seguito è necessario creare finalmente il modello in spazio di stato utilizzando il comando **ss**, che riceve in input proprio i parametri prima definiti:

```
spazio_stato = ss (F,G,H,J); %Crea il modello in spazio di
                             %stato
```

1.3 Funzione di trasferimento del modello

Una volta ricavato il modello in spazio di stato, si può passare alla derivazione della funzione di trasferimento, calcolando la trasformata di Laplace dell'equazione differenziale in (1.1), assumendo pari a zero le condizioni iniziali. Ottengo:

$$U(s) = MsV(s) + bV(s) \tag{1.4}$$

Ricordando che l'uscita del sistema è la velocità, la funzione di trasferimento è allora:

$$G(s) = \frac{V(s)}{U(s)} = \frac{1}{Ms + b} \tag{1.5}$$

1.3.1 Implementazione della funzione di trasferimento in ambiente *Matlab*

```
s = tf('s');       %Creazione di una funzione di trasferimento
                   %sotto il simbolo 's'
F_t= 1/(M*s+b);    %Definisco la funzione di trasferimento
```

1.4 Specifica delle prestazioni

Lo step successivo è quello di definire alcune specifiche di progetto che devono essere rispettate, al fine di ottenere una corretta simulazione del sistema di controllo.
Dai parametri fisici che abbiamo assegnato al sistema, abbiamo osservato che, quando il motore fornisce una forza di 500 N, la vettura raggiunge una velocità massima di 10 m/s (22 mph – 36 km/h) e, in condizioni usuali, dovrebbe essere in grado di raggiungere questa velocità in non più di 5 secondi.
Per quest'applicazione, è sufficiente una sovraelongazione della velocità del 10% e un errore a regime del 2%.
Tenendo in considerazione quanto appena detto, possiamo definire le seguenti specifiche:

- Tempo di salita[1] (Rise Time) < 5 s
- Sovraelongazione[2] (Overshoot) $< 10\%$
- Errore a regime[3] (Steady-State Error) $< 2\%$

[1] Indica il tempo necessario al sistema per variare dal 10% al 90% del valore di regime dello stesso.

[2]Indica la differenza tra il valore di output a regime e il massimo valore di output raggiunto durante il transitorio.

[3] Indica la variazione percentuale tra il valore di output a regime e il valore desiderato.

CAPITOLO 2
Analisi dinamica del sistema in catena aperta

2.1 Risposta al gradino

Dato un modello input/output, definiamo la sua *risposta al gradino* (*step response)*, $w_{-1}(t),\ con\ t \in \mathbb{R}$, la risposta forzata al segnale di ingresso $u(t) = \partial_{-1}(t)$.

La risposta al gradino del sistema in questione, senza alcun controllo di retroazione, in corrispondenza di un ingresso a gradino di ampieza 500 N, può essere visualizzata con il seguente comando *Matlab*:

```
step(u*F_t);            %Risposta al gradino di ampiezza 500 N
```

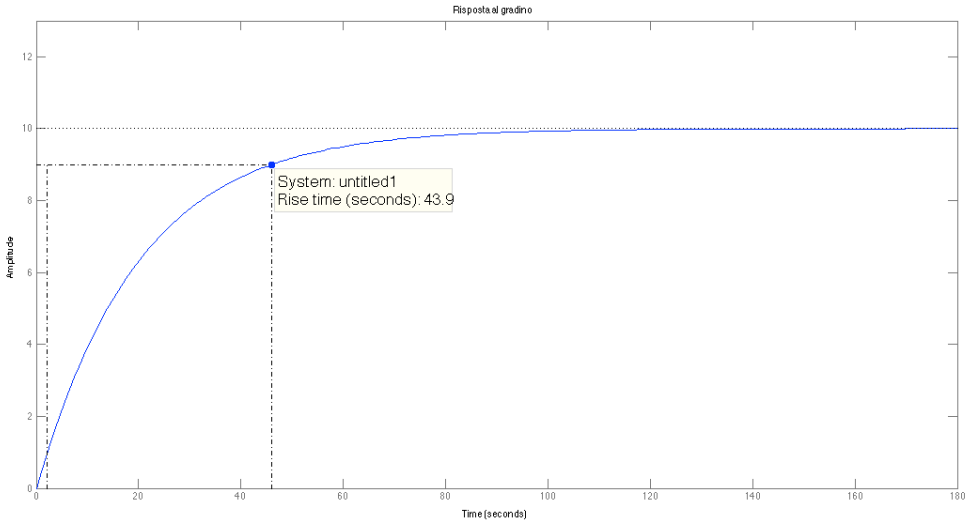


Figura 2.1: Risposta al gradino in assenza di retroazione (ad anello aperto)

Vediamo come il sistema non retroazionato non presenta oscillazioni o sovraelongazioni (com'è tipico per i sistemi del primo ordine) e riesce anche a

raggiungere la velocità di regime di 10 m/s.
Tuttavia il tempo di salita (*rise time*) è troppo alto (43.9 s). Abbiamo quindi bisogno di progettare un controllore in retroazione che accelera significativamente la risposta al gradino, senza influire negativamente sulle altre prestazioni dinamiche.

2.2 Analisi di poli e zeri ad anello aperto

Il sistema del controllo della velocità di crociera (*cruise control*) presenta un unico polo in corrispondenza del punto $-\frac{b}{M}$, come può essere ricavato dall'equazione (1.5), trovando le soluzioni del denominatore della funzione di trasferimento.
Possiamo vedere graficamente la posizione di tale polo sul piano s usando i seguenti comandi *Matlab*:

```
pzmap(F_t);           % Rappresenta poli e zeri della FdT
 axis ([-1 1 -1 1]) % Definisce gli estremi dell'asse Re e Im
```

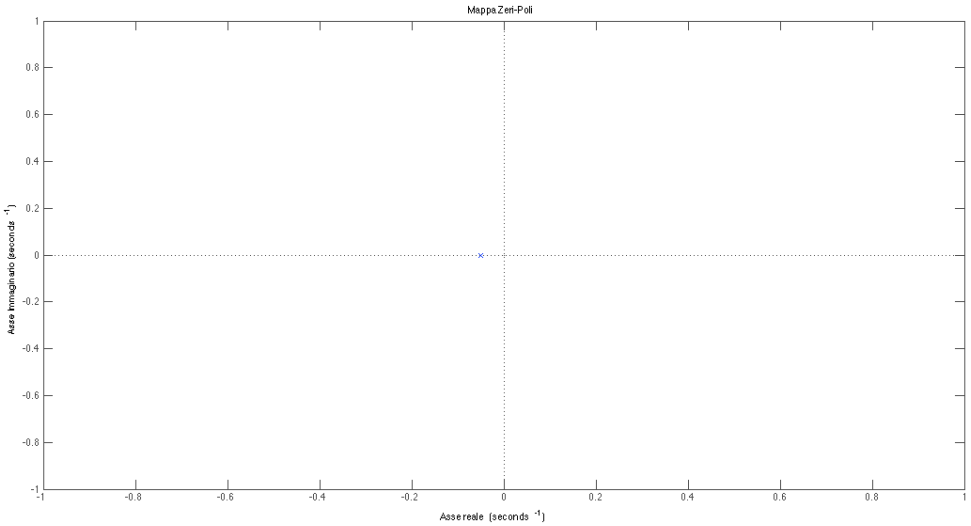


Figura 2.2: Grafico dei poli e zeri ad anello aperto nel piano complesso

Osserviamo che il sistema ad anello aperto è *BIBO-stabile*[4] e privo di oscillazioni, dato che l'unico polo è a parte reale negativa.

[4] Non possiamo dire nulla sulla stabilità asintotica poiché stiamo considerando una funzione di trasferimento, nella quale possono esserci state cancellazioni zero-polo di componenti instabili, che adesso non saremmo più in grado di determinare.

Inoltre la velocità della risposta è determinata dal valore di questo polo: maggiore è il suo valore, più rapidamente il sistema si avvicina al suo valore di regime. Poiché il polo è molto piccolo ($\sim$ -0.05), questo sistema è particolarmente lento, come abbiamo già avuto modo di osservare dalla figura 2.1. Dal momento che, in generale, non siamo in grado di modificare i parametri del sistema per modificarne la sua risposta dinamica, siamo costretti a dover ricorrere alla progettazione di controllori che, alterando la posizione dei poli e degli zeri nel sistema, questa volta retroazionato (ad *anello chiuso*), permettono di ottenere le specifiche di progetto richieste.

2.3 Diagrammi di Bode ad anello aperto

Possiamo provare adesso a determinare la risposta in frequenza ad anello aperto del sistema.
Ciò può essere visto graficamente grazie ai *diagrammi di Bode* delle ampiezze e delle fasi, che possono essere ottenuti con i seguenti comandi *Matlab*:

```
bode (F_t);      %Traccia i diagrammi di Bode di modulo e fase
```

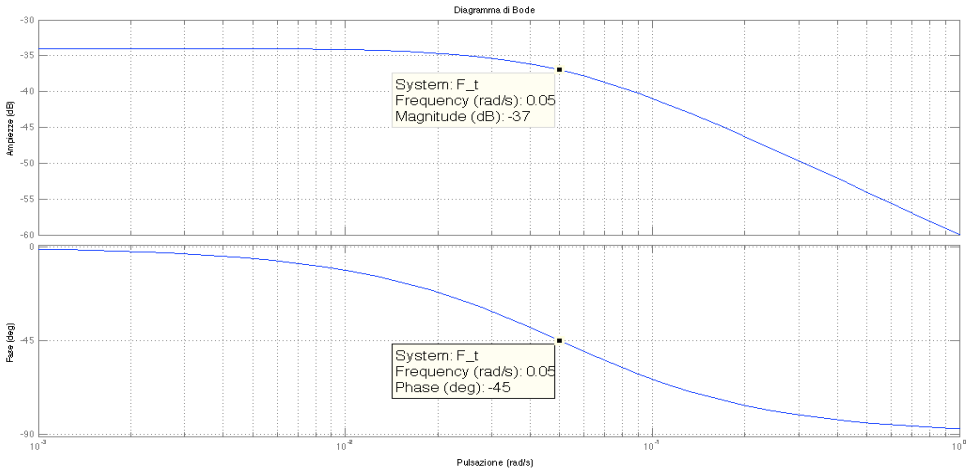


Figura 2.3: Diagramma di Bode di modulo e fase

E' possibile tracciare i diagrammi di Bode visualizzando simultaneamente anche quelli asintotici con i seguenti comandi *Matlab*:

```
bodeasint (1,[M,b],0.001,1,'mod',0,0);    %Diagramma di Bode delle
                                           %ampiezze
```

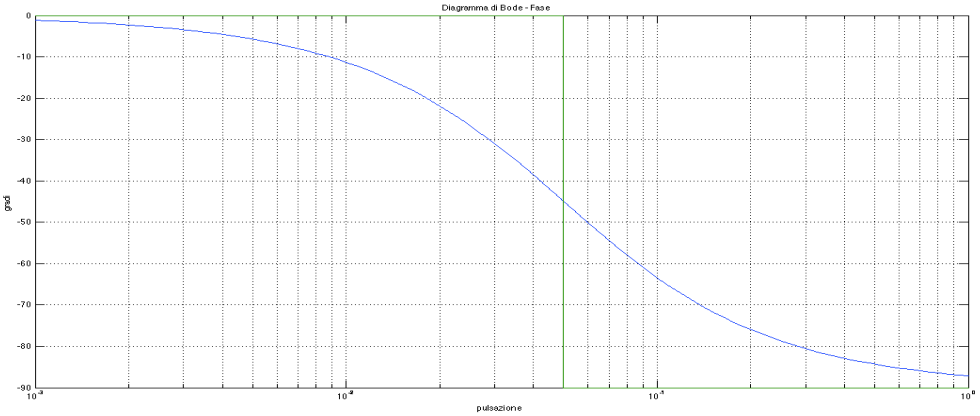


Figura 2.4: Diagramma di Bode del modulo (con componente asintotica)

```
bodeasint (1,[M,b],0.001,1,'arg',0,0);    %Diagramma di Bode delle
                                          %fasi
```

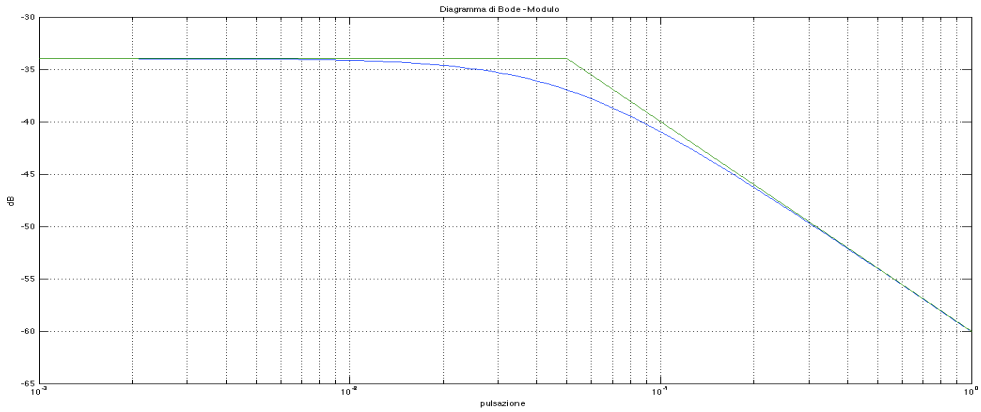


Figura 2.5: Diagramma di Bode della fase (con componente asintotica)

Possiamo osservare dai diagrammi di Bode che, in corrispondenza della pulsazione $\omega = -\frac{b}{M} = 0.05$, il modulo è sceso di 3dB (partiva con un guadagno di -34dB) e la fase corrispondente è scesa a -45°.
Infine, alle frequenze elevate, il diagramma di Bode delle ampiezze scende infinitamente con una pendenza di -20dB/dec.

CAPITOLO 3
Progetto del sistema di controllo

3.1 Panoramica sulla progettazione del controllore

Un sistema di controllo è progettato per una determinata applicazione e le caratteristiche che esso deve possedere dipendono ovviamente da esigenze particolari. In particolare si cerca, attraverso un parametro di controllo C(s), di far sì che lo schema in retroazione (ad *anello chiuso*) fruisca di caratteristiche migliorative che lo schema non retroazionato non possedeva. Possiamo elencare alcune proprietà che abitualmente si cerca di conseguire con il controllo:

1. **Errore a regime** specifico nella risposta del sistema ai segnali canonici;

2. **Stabilizzazione** di un sistema che potrebbe essere instabile;

3. **Prontezza del sistema**, cercando di ridurre il tempo di salita nella risposta al gradino;

4. **Capacità smorzante,** che è tanto maggiore quanto più è piccola la sovraelongazione nella risposta al gradino;

5. **Insensibilità alle variazioni parametriche e ai disturbi**

Nel nostro esempio specifico, come già detto, dobbiamo cercare di ridurre il più possibile il tempo di salita del sistema.
Dato che stiamo considerando un sistema del primo ordine, esso è rappresentabile schematicamente nel modo che segue:

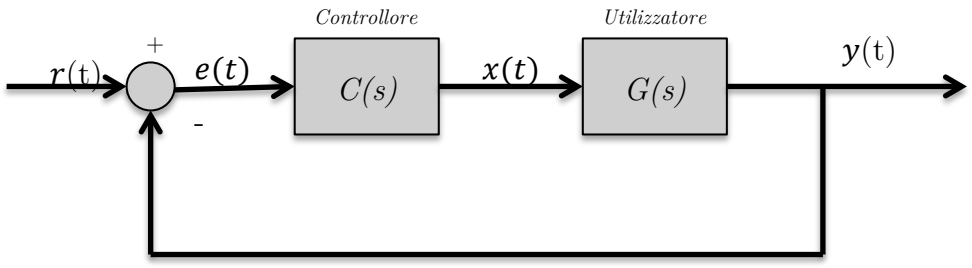


Figura 3.1: Schema a blocchi del sistema con retroazione unitaria negativa

La retroazione nello schema della figura 3.1 è *unitaria* perché l'uscita è riportata invariata all'ingresso e *negativa* perché, per convenzione, si suppone l'uscita sottratta all'ingresso, nella retroazione. Nello schema:

- C(s) è la funzione di trasferimento del controllore del sistema. Tale compensatore che deve essere progettato opportunamente;
- G(s) è la funzione di trasferimento in catena aperta, cioè la (1.5);
- r(t) è l'ingresso esterno di riferimento;
- e(t) è il segnale r(t) – y(t) e rappresenta l'errore che il sistema compie per far assumere all'uscita il valore che esso assume all'ingresso, cioè r(t);
- y(t) è l'uscita del sistema.

Possiamo unire i due blocchi C(s) e G(s) in un unico blocco W(s)=C(s)G(s), sfruttando le proprietà della trasformata di Laplace.

Ci chiediamo adesso in che modo la retroazione modifichi G(s) e quale sia la funzione di trasferimento che otteniamo.
Questo può essere facilmente dedotto dai semplici calcoli che seguono. Ricorriamo alla trasformata di Laplace per i segnali e(t),r(t) e y(t).

$$E(s) = R(s) - Y(s) \tag{3.1}$$

Utilizzando le proprietà di Laplace note e osservando la figura 3.1, possiamo ricavare

$$X(s) = E(s)C(s) = \big(R(s) - Y(s)\big)C(s) \tag{3.2}$$

$$\begin{aligned} Y(s) &= X(s)G(s) = \big(R(s) - Y(s)\big)C(s)G(s) \\ &= R(s)C(s)G(s) - Y(s)C(s)G(s) \end{aligned} \tag{3.3}$$

Infine, dalla (3.3), siamo ora in grado di ricavare la funzione di trasferimento del sistema retroazionato:

$$\boxed{H(s) = \frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{W(s)}{1 + W(s)}} \tag{3.4}$$

3.2 Controllori PID

Il controllore PID è un tipo di compensatore che, nonostante la sua semplicità, è molto utilizzato perché:

1. Funziona correttamente "quasi sempre", per problemi di controllo standard;

2. Si preferisce usare un meccanismo di controllo tanto semplice quanto sicuro, piuttosto che avventurarsi nella progettazione di compensatori più efficienti ma più costosi e dall'esito più incerto.

La funzione di trasferimento di un controllore PID è:

$$C_{PID} = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s} \tag{3.5}$$

Osservando la (3.5), possiamo costatare come il PID sia formato dalla somma di tre componenti:

1. **P→PROPORZIONALE:** C(s)=K_p: ha l'effetto di ridurre il tempo di salita (che è ciò che vogliamo ottenere) e riduce ma non elimina l'errore a regime;

2. **I→INTEGRALE:** C(s)=K_i/s: ha l'effetto di eliminare completamente l'errore a regime per un ingresso costante o per il gradino unitario, ma può rendere la risposta transitoria più lenta;

3. **D→DERIVATIVO:** $C(s)=K_d s$: ha l'effetto di aumentare la stabilità del sistema, riducendo la sovraelongazione e migliorando la risposta transitoria.

Gli effetti di ciascun parametro del PID sono riassunti in questa tabella[5]:

Parametro	T. salita	Sovraelong.	T. assestam.	E. regime
K_p	Scende	Aumenta	Piccola variazione	Scende
K_i	Scende	Aumenta	Aumenta	Eliminato
K_d	Piccola Variazione	Diminuisce	Diminuisce	Invariato

Dal compensatore PID possano essere ottenuti dei controllori più semplici, variando i valori dei suoi parametri fondamentali: il controllore **PI** (per $K_d=0$), il controllore **PD** (per $K_i=0$), il controllore **I** (per $K_d=K_p=0$), il controllore **P** (per $K_i=K_d=0$).

ATTENZIONE: il controllore PID non è fisicamente realizzabile perché, come si evince dalla (3.5), stiamo considerando una funzione di trasferimento *impropria*. Nella pratica è allora necessario introdurre dei poli in alta frequenza che non modificano il funzionamento del controllore ma che altresì rendono propria la FdT, affinché possa essere utilizzata per i nostri scopi.

3.2.1 Implementazione del PID in *Matlab*

Possiamo definire un controllore PID in ambiente *Matlab* usando direttamente la funzione di trasferimento (3.5) con i seguenti comandi:

```
Kp = 1;                          %Costante proporzionale
Ki = 1;                          %Costante integrale
Kd = 1;                          %Costante derivativa
s = tf ('s');
C = Kp + Ki/s + Kd*s    %Definisco manualmente la FdT
```

[5] Questa tabella è puramente indicativa: le correlazioni tra parametri non sono così accurate, poiché questi sono dipendenti gli uni dagli altri; cambiando il valore di una di queste variabili, può cambiare l'effetto complessivo delle altre due sull'intero sistema.

In alternativa, si può usare il comando presente nella libreria *Matlab* che genera automaticamente un controllore PID, nel momento in cui sono forniti i suoi parametri:

```
C = pid (Kp, Ki, Kd)        %Realizza automaticamente un PID
```

3.2.2 Implementazione controllore P (Proporzionale)

Torniamo a considerare il sistema in retroazione della figura 3.1; dobbiamo cercare di trovare una funzione di trasferimento per il sistema ad anello chiuso, aggiungendo un controllore proporzionale $C(s)=K_p$ (abbiamo infatti sottolineato come un compensatore di questo tipo permetta proprio di ridurre il tempo di salita del sistema).

Nella (3.4), sostituiamo il valore di C(s) di tipo **P** e della FdT espressa dalla (1.5). Otteniamo in seguente risultato, che costituisce l'equazione 3.6:

$$H(s) = \frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{K_p \frac{1}{Ms + b}}{1 + K_p \frac{1}{Ms + b}} = \frac{K_p}{Ms + b + K_p}$$

Possiamo adesso costruire il grafico della nuova risposta al gradino, assumendo $K_p=100$ e con una velocità di riferimento di 10 m/s. In un nuovo file di tipo .m, i comandi *Matlab* sono i seguenti:

```
M = 1000;        %massa del veicolo
b = 50;          %coefficiente complessivo di attrito
r = 10;          %velocità in m/s assunta costante (è l'ingresso)
s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento G(s)
Kp = 100;               %costante del compensatore P
C = pid (Kp);   %crea un controllore PID, con il solo parametro Kp
Fb = feedback (C * F_t, 1)    %collega più blocchi in un unico
                               %sistema in retroazione
t = 0:0.1:20;
step (r * Fb, t)
axis ([0 20 0 10])
```

Attraverso questi comandi, si ottiene la funzione di trasferimento $T = \frac{100}{1000s+150}$ e nella figura 3.2 riportata qui sotto si può osservare il grafico della risposta al gradino per tale funzione di trasferimento:

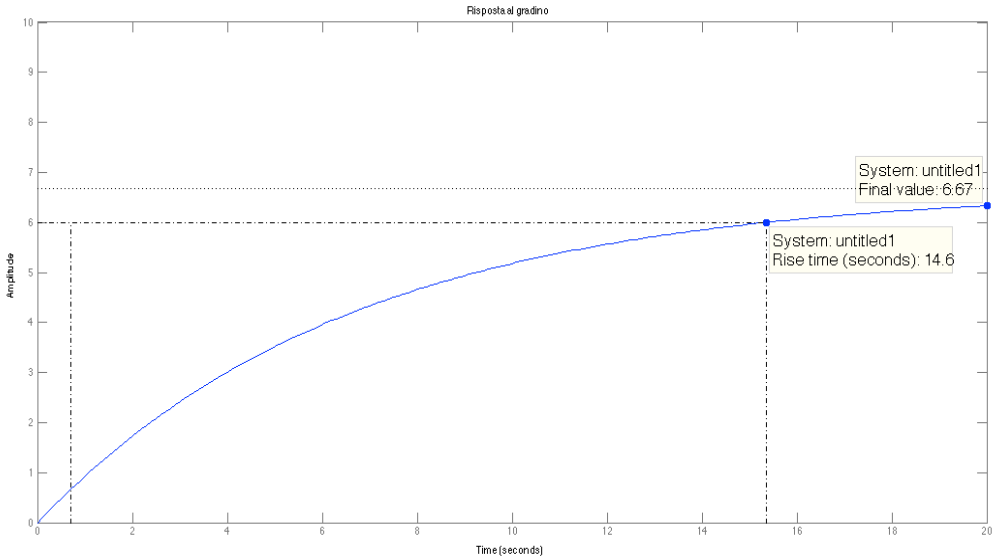


Figura 3.2: Risposta al gradino con controllore tipo **P** con K_p=100

Vediamo adesso di confrontare questo grafico con quello della figura 2.1. Sebbene, grazie all'inserimento del compensatore proporzionale, la risposta al gradino risulti più rapida rispetto a quello della figura 2.1 (14.6 secondi contro 43.9 secondi), né l'errore a regime né il tempo di salita soddisfano ancora i nostri criteri di progettazione.[6]
Una possibile soluzione è di aumentare il guadagno proporzionale, K_p, per ridurre ulteriormente il tempo di salita e l'errore a regime. Possiamo modificare il file .m precedente in modo da porre K_p=5000; questo è il grafico della risposta al gradino che otteniamo, rappresentato nella figura 3.3.

Osservando la figura 3.3, possiamo vedere come l'errore a regime sia in pratica ormai nullo (riusciamo ad arrivare, a regime, fino a 9.9 m/s, cioè a ridosso della nostra velocità richiesta) e come il tempo di salita si sia ridotto considerevolmente. Tuttavia questa risposta è irrealistica perché un vero sistema di *cruise control* non può essere in grado di far variare la velocità di un veicolo da 0 m/s a 9 m/s in circa 0.44 secondi, a causa di ovvie limitazioni della potenza del motore di cui l'automezzo è fornito.

[6] Il *rise time* vale 14.6 secondi, ancora inferiore ai 5 secondi che il problema richiede, mentre il sistema a regime arriva a 6.67 m/s, ben al di sotto dell'errore del 2% che possiamo concederci.

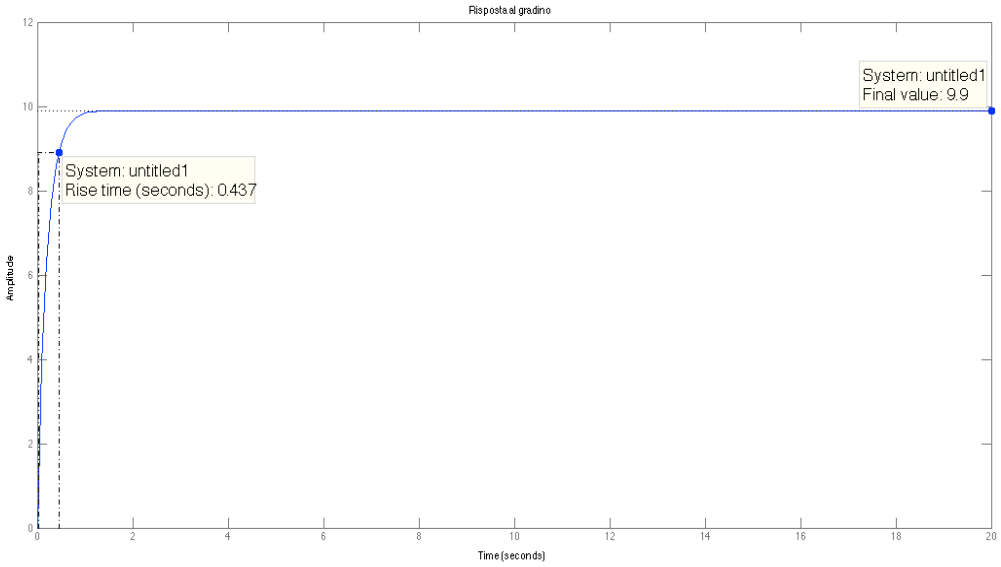


Figura 3.3: Risposta al gradino con controllore tipo **P** con K_p=5000

La soluzione sarà allora quella di ridurre il guadagno proporzionale K_p, a costo di un aumento, pur ragionevole, del tempo di salita, e aggiungere un controllore integrale (di tipo **I**) per eliminare definitivamente l'errore a regime per l'ingresso a gradino.

3.2.3 Implementazione controllore PI (Proporzionale-Integrale)

Abbiamo già visto nella sezione 3.2.1 che il controllore PI può essere ottenuto a partire da quello PID semplicemente annullando il parametro K_d. Di conseguenza la sua funzione di trasferimento è:

$$C_{PI} = K_p + \frac{K_i}{s} \tag{3.7}$$

Sostituendo nell'equazione (3.4) il valore di C(s), risulta:

$$H(s) = \frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{\left(K_p + \frac{K_i}{s}\right)\frac{1}{Ms + b}}{1 + \left(K_p + \frac{K_i}{s}\right)\frac{1}{Ms + b}} = \frac{K_p s + K_i}{Ms^2 + \left(b + K_p\right)s + K_i} \tag{3.8}$$

Come prima, possiamo passare a realizzare il grafico della nuova risposta al gradino, assumendo K_p=600 e K_i =1. Modifichiamo il file di tipo .m con i seguenti comandi *Matlab*:

```
M = 1000;        %massa del veicolo
b = 50;          %coefficiente complessivo di attrito
r = 10;          %velocità in m/s assunta costante
s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento G(s)
Kp = 600;        %costante proporzionale P
Ki=1             %costante integrale I
C = pid (Kp,Ki);%controllore PID, con i soli parametri Kp, Ki
Fb = feedback (C * F_t, 1)  %collega più blocchi (FdT) in un
                             %unico sistema in retroazione
step (r * Fb, t)
```

Il grafico della risposta al gradino è:

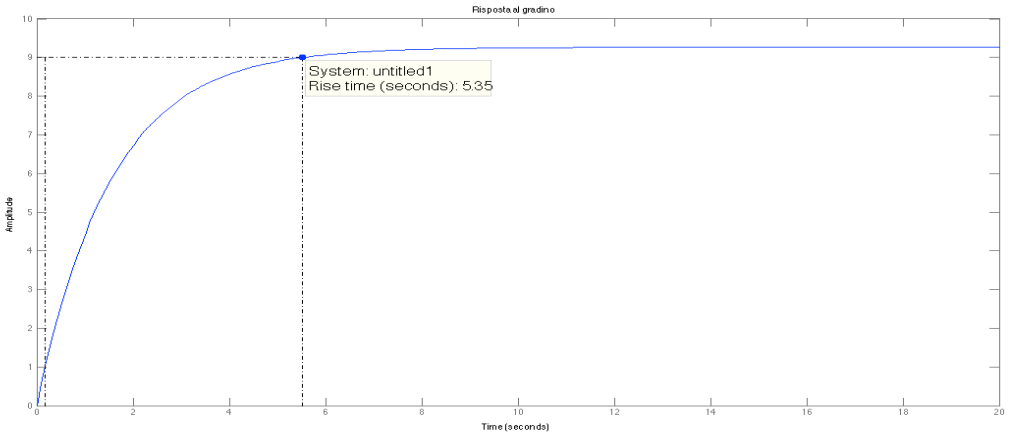
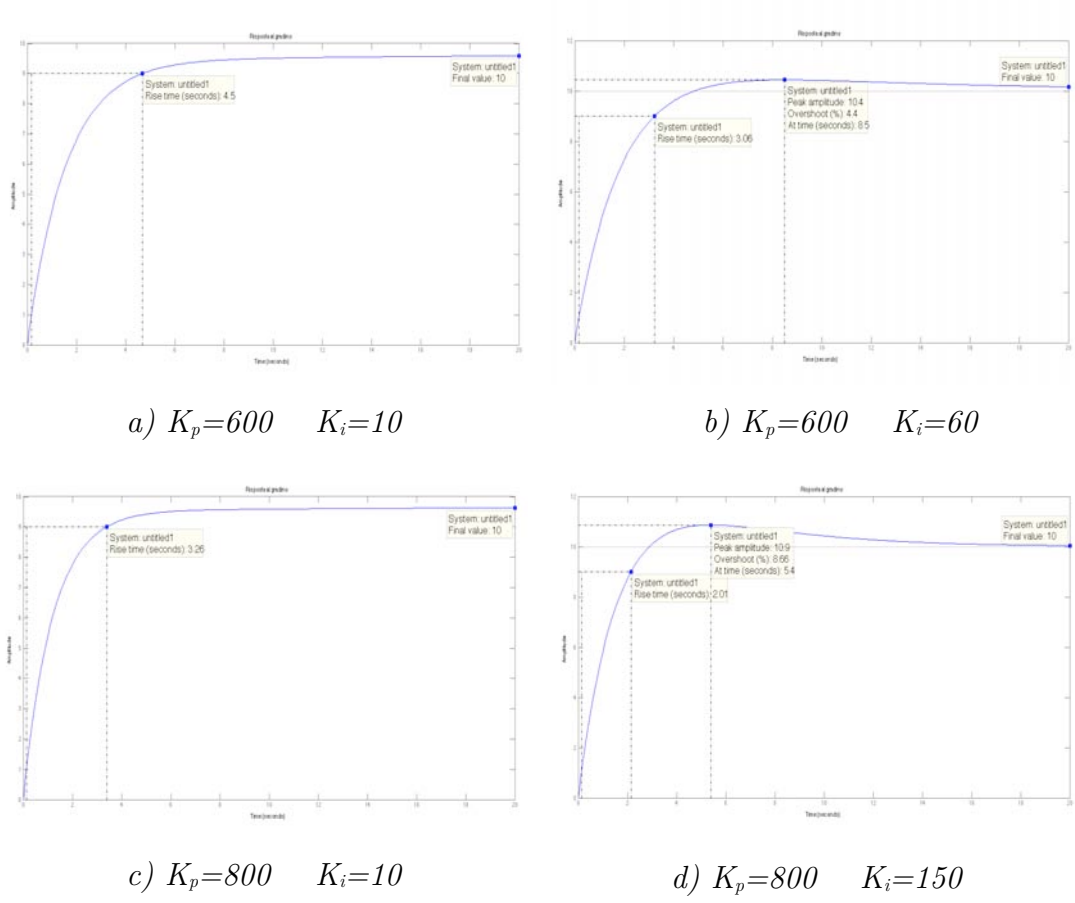


Figura 3.4: Risposta al gradino con controllore tipo **PI** con K_p=600 e K_i=1

Dalla figura 3.4, si osserva che le specifiche non sono rispettate ed è necessario pertanto modificare i valori dei parametri K_p e K_i del file .m, fino a quando non si ottiene la risposta desiderata, facendo attenzione alle seguenti osservazioni:

- Un guadagno proporzionale troppo elevato potrebbe rendere il sistema non fisicamente realizzabile (come già visto nella sezione precedente);
- Un guadagno integrale troppo elevato può rendere instabile la risposta al gradino, presentando una sovraelongazione indesiderata.

Nella figura 3.5 sono rappresentate quattro possibili scelte per i valori dei parametri del controllore, con le quattro rispettive risposte al gradino.



a) K_p=600 K_i=10 *b)* K_p=600 K_i=60

c) K_p=800 K_i=10 *d)* K_p=800 K_i=150

Figura 3.5: Risposte al gradino con controllore tipo **PI** al variare dei parametri K_p e K_i

In accordo con la tabella riassuntiva del paragrafo 3.2.1, possiamo osservare come, all'aumentare del parametro K_i, si registri una diminuzione del tempo di salita ma un aumento della sovraelongazione (come si può vedere nella figura 3.5 a) e 3.5 b)).
Se aumentiamo il parametro K_p (figure c) e d)), stiamo migliorando il *rise time*, mantenendolo entro i limiti richiesti e su valori fisicamente accettabili.
Osserviamo infine come, aumentando il parametro K_i fino al valore di 150, pur diminuendo il tempo di salita, ci imbattiamo in una sovraelongazione del 9%, quasi al limite delle nostre specifiche.
Infine, a causa della presenza del compensatore integrale, garantiamo che l'errore a regime sia sempre identicamente nullo.

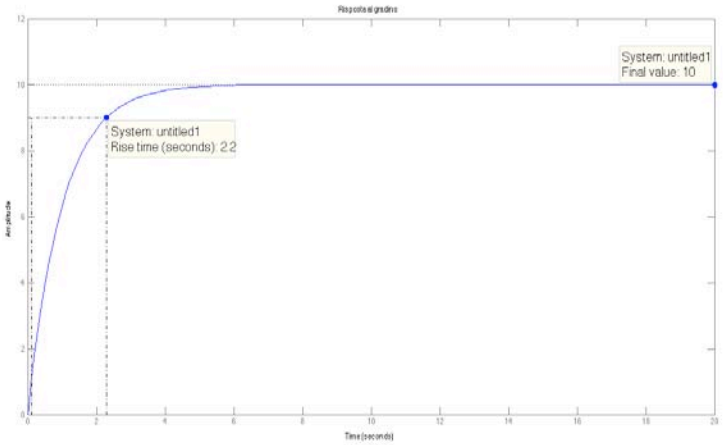


Figura 3.6: Risposta al gradino con controllore tipo **PI** con K_p=1000 e K_i=50

Un'implementazione ottimale si raggiunge ponendo K_p=1000 e K_i=50: con questi parametri il tempo di salita è di 2.2 secondi (<5 s richiesti), e la sovraelongazione e l'errore a regime sono nulli.

3.2.4 Implementazione controllore PID (Proporzionale-Integrale-Derivativo)

Per quanto riguarda questo esempio specifico, al fine di conseguire gli obbiettivi di progetto richiesti, non è necessario ricorrere alla progettazione di un controllore PID, in quanto già un'opportuna scelta dei parametri K_p e K_i di un PI fornivano dei risultati soddisfacenti.
Tuttavia ci si chiede come varia la risposta al gradino del sistema, inserendo nel compensatore anche una componente derivativa che, come sappiamo, consente di ridurre la sovraelongazione.
Dalla (3.5) e sostituendo la (3.4) a C(s), otteniamo la funzione di trasferimento:

$$H(s) = \frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1 + C(s)G(s)} = \frac{K_d s^3 + K_p s + K_i}{(M + K_d)s^2 + (b + K_p)s + K_i} \tag{3.9}$$

Per ottenere il grafico della risposta in frequenza, creiamo un nuovo file .m con i comandi *Matlab* riportati sotto, facendo variare i parametri K_p, K_i e K_d, fino a raggiungere gli obbiettivi richiesti (i parametri nel codice Matlab sottostante sono arbitrari):

```
M = 1000;        %massa del veicolo
b = 50;          %coefficiente complessivo di attrito
r = 10;          %velocità in m/s assunta costante

s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento G(s)
Kp=800;        %costante proporzionale P
Ki=10;            %costante integrale I
Kd=10;             %costante derivativa D
C = pid (Kp,Ki,Kd);   %crea un controllore PID
Fb = feedback (C * F_t, 1)    %collega più blocchi (FdT) in
                              %un unico sistema in retroazione
t = 0:0.1:20;
step (r*Fb,t);
axis ([0 20 0 10])
```

Dopo vari tentativi, deduciamo che una possibile combinazione di parametri per cui siano rispettate le specifiche di progetto sul tempo di salita, sovraelongazione ed errore a regime è con K_p=1000, K_i=45, K_d=20, come è possibile osservare dalla risposta al gradino qui riportata:

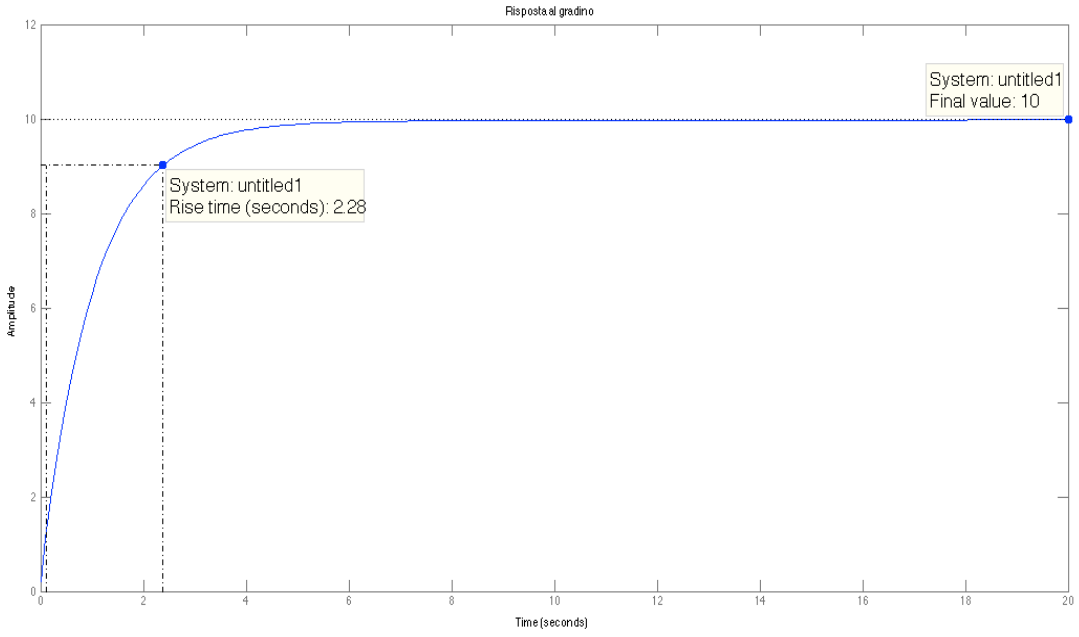


Figura 3.7: Risposta al gradino con controllore tipo **PID** con Kp=1000, Ki=45, Kd=20

3.2.5 Considerazioni finali

Come già osservato precedentemente anche con l'ausilio di *Matlab*, l'uso di un solo compensatore proporzionale **P** non è sufficiente per risolvere le specifiche del problema, dato che, con un aumento eccessivo di K_p al fine di rendere il sistema più rapido, si incorrerebbe in problemi di irrealizzabilità fisica.
Tuttavia, semplicemente aggiungendo una componente integrale **I**, in seguito alla scelta di opportuni parametri, siamo già in grado di ottenere delle soluzioni più che accettabili al nostro problema.
Abbiamo comunque provato a utilizzare un controllore **PID**, osservando che la risposta al sistema retroazionato risulta essere altrettanto stabile e rapida quanto quella ottenuta con il compensatore di **PI**, il cui uso è quindi auspicabile rispetto al primo per garantire una maggiore semplicità ed elementarità del sistema.

3.3 Analisi con il luogo delle radici

3.3.1 Introduzione al luogo delle radici

Il metodo del luogo delle radici (*root locus*) è un procedimento grafico usato per studiare il movimento dei poli di una funzione di trasferimento ad anello chiuso al variare di un parametro reale K, al fine di studiarne le proprietà, come ad esempio la stabilità.
Sebbene esistano altre tecniche con le quali è possibile analizzare un sistema (ad esempio con il *criterio di Routh* o con il *criterio di Nyquist*), il metodo del luogo delle radici dà informazioni più dettagliate sulla natura dei poli della funzione in retroazione, per certi valori di K, (come la presenza di fenomeni oscillatori o di poli complessi) rispetto a quelle che possono fornire gli altri metodi sopra elencati.
Lo svantaggio è che il metodo del luogo delle radici è molto più sensibile e quindi meno robusto alle variazioni parametriche che possono esserci nel sistema rispetto, ad esempio, al criterio di Nyquist.

Riprendendo la (3.4), avevamo trovato che la funzione di trasferimento di un sistema in retroazione è $H(s) = \frac{Y(s)}{R(s)} = \frac{C(s)G(s)}{1+C(s)G(s)} = \frac{W(s)}{1+W(s)}$.

Scegliendo C(s)=K ed utilizzando come FdT $G(s) = \frac{b(s)}{a(s)}$, l'equazione (3.4) diventa:

$$H(s) = \frac{Kb(s)}{a(s) + Kb(s)} \tag{3.10}$$

Per studiare, ad esempio, la stabilità del sistema, sarà necessario analizzare la natura dei poli della (3.10) e quindi risolvere l'equazione

$$a(s) + Kb(s) = 0 \tag{3.11}$$

al variare del parametro K. Tramite il metodo del luogo delle radici è quindi possibile conoscere la locazione di tutti i possibili poli ad anello chiuso, quando varia il valore di K.

3.3.2 Controllore proporzionale con *root locus*

Dato che il metodo del luogo delle radici è basato sulla variazione di un parametro K, è lecito pensare che esso possa essere facilmente implementato da un controllore proporzionale (tipo **P**) con guadagno K_p=K.
Sostituendo nella (3.4) il valore di C(s)=K_p e la G(s) della (1.5), otteniamo la seguente FdT:

$$H(s) = \frac{K_p}{Ms + (K_p + b)} \tag{3.12}$$

Prima di passare ai codici *Matlab* che ci consentiranno di visualizzare graficamente il luogo delle radici, è necessario determinare prima sia il coefficiente di smorzamento (ξ) sia la pulsazione naturale (ω_n).
Detto T_r il tempo di salita (*rise time*) e M_p la massima sovraelongazione, per ricavare i due parametri richiesti saranno utilizzate le seguenti equazioni:

- $\omega_n \geq \frac{1.8}{T_r}$: poiché vogliamo che il tempo di salita sia inferiore ai 5 secondi, da questa equazione deduciamo che dobbiamo avere una pulsazione naturale maggiore di 0.36.

- $\xi \geq \sqrt{\frac{ln^2(M_p)}{\pi^2 + ln^2(M_p)}}$: poiché abbiamo imposto un limite del 10% alle sovraelongazioni, da questa equazione deduciamo che il coefficiente di smorzamento deve obbligatoriamente essere maggiore di 0.6.

A questo punto è possibile creare un nuovo file .m con i seguenti comandi *Matlab*:

```
M = 1000;         %massa del veicolo
b = 50;           %coefficiente complessivo di attrito
r = 10;           %velocità in m/s assunta costante
s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento G(s)
rlocus (F_t)         %comando per il luogo delle radici
axis ([-0.6 0 -0.6 0.6]);
sgrid (0.6, 0.36)        % valori dello smorzamento e della
                         % pulsazione naturale
```

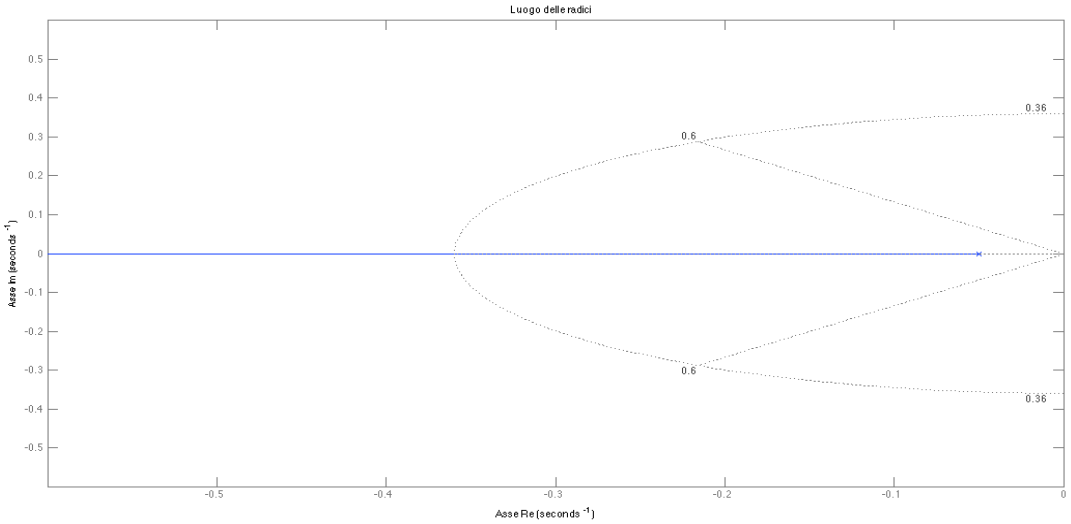


Figura 3.8: Grafico del luogo delle radici

Dalla figura 3.8 possiamo osservare la presenza di due linee tratteggiate[7]:

- la semi-ellisse rappresenta tutti i punti del luogo in cui la pulsazione naturale vale esattamente 0.36: ω_n è maggiore di 0.36 al di fuori di tale confine, mentre è più piccola al suo interno;

- le due semirette stabiliscono tutti i punti del luogo in cui il coefficiente di smorzamento vale esattamente 0.6: ξ è maggiore di 0.6 nell'area compresa all'interno delle due linee, mentre è minore all'esterno;

[7] Queste linee compaiono poiché abbiamo fornito il comando "*sgrid*".

A questo punto esiste il comando *Matlab "rlocfind"* che permette di posizionare un polo (facendo variare il valore di ω_n) in un punto a nostra scelta sul luogo; quindi restituisce il valore del guadagno K_p che il sistema ad anello chiuso dovrebbe avere per presentare un polo nel punto scelto. Il comando è:

```
[Kp, poles] =rlocfind (F_t) %sceglie il punto per il polo
```

Dopo l'esecuzione dell'istruzione nella finestra di comando, ci è permesso di selezionare un punto a nostra scelta sul luogo delle radici. Dal momento che vogliamo avere un polo all'interno dell'area compresa tra le linee tratteggiate (ξ>0.6) ma al di fuori della semi-ellisse (ω_n>0.36), è corretto scegliere un punto sull'asse reale appena fuori dalla semi-ellisse (~ -0.5), come indicato dalla croce rossa nella figura sottostante:

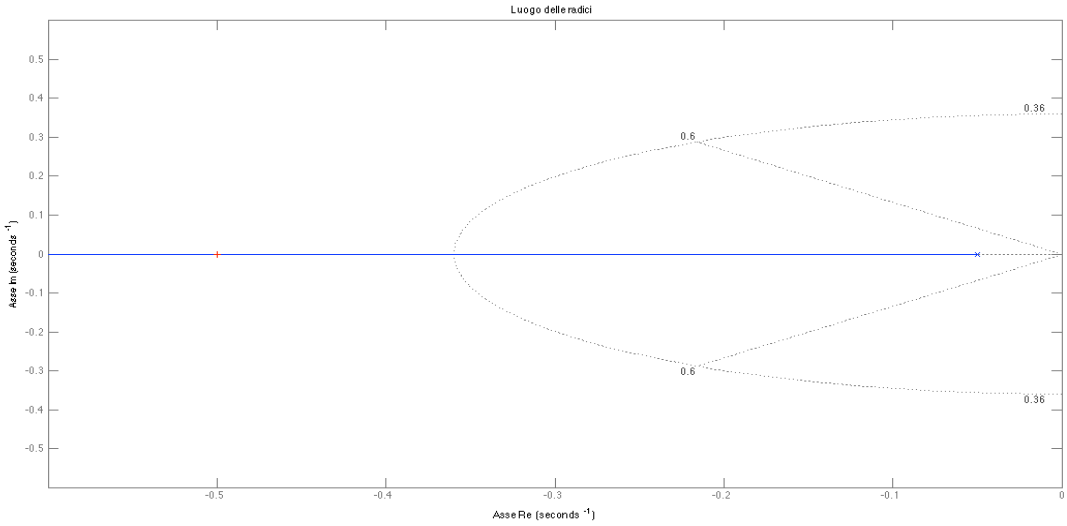


Figura 3.9: Grafico del luogo delle radici con scelta del polo

Matlab fornirà quindi i seguenti risultati:

```
Select a point in the graphics window

selected_point =
-0.4999 - 0.0012i
      Kp =
   449.8996
   poles =
   -0.4999
```

Per avere un polo con le caratteristiche richieste, quindi, il nostro controllore dovrebbe avere un guadagno K_p di circa 450.
Possiamo generare allora la risposta al gradino per un sistema ad anello chiuso, il cui compensatore ha una costante proporzionale pari al valore restituito precedentemente, con questi comandi *Matlab*:

```
Kp = 449.8996;          %Impone il valore per la il guadagno
Fb = feedback (Kp * F_t, 1)    %collega più blocchi (FdT) in
                               %un unico sistema in retroazione
t = 0:0.1:20;
step (r * Fb, t)
```

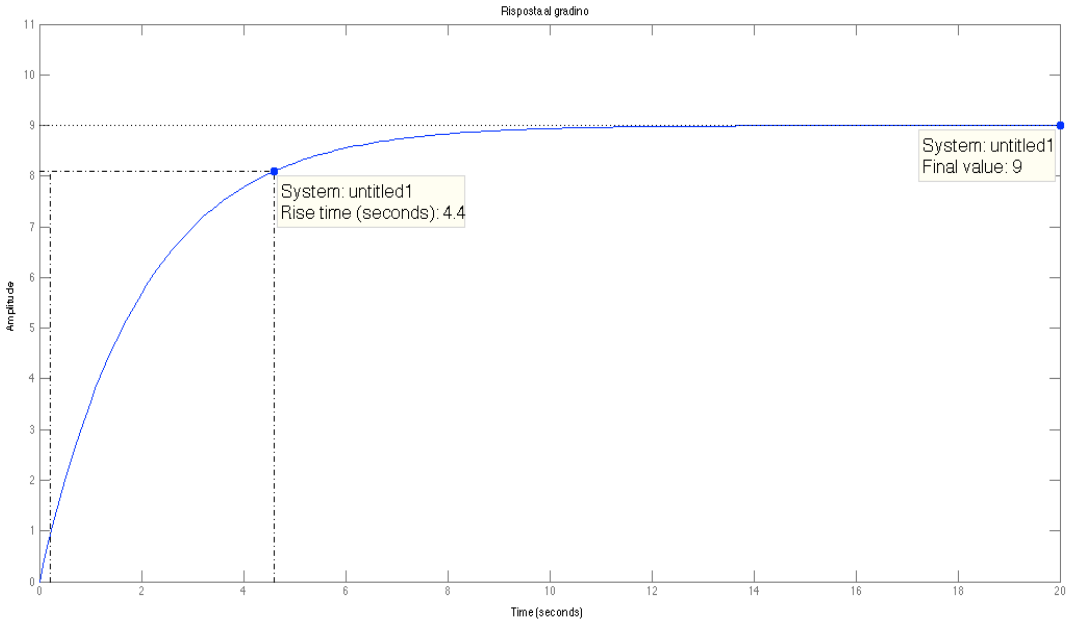


Figura 3.10: Risposta al gradino con K_p ottenuto automaticamente

Notiamo che, con il guadagno K_p appena scelto, i vincoli sul tempo di salita (<5 s) e sulla sovraelongazione (<10 %) sono rispettati; al contrario, permane un errore a regime ben al di sopra delle esigenze di progetto (10%, contro il 2% desiderato).

Possiamo comunque osservare dalle figure sottostanti come varia la risposta al gradino al variare del posizionamento del polo sul luogo delle radici e quindi al variare della costante proporzionale del sistema.

Come ci aspettavamo, sulla base delle conoscenze sui controllori di tipo **P**, a mano a mano che il punto sull'asse reale si allontana dalla semi-ellisse, il guadagno proporzionale K_p cresce gradatamente, con la conseguente diminuzione del tempo di salita e il miglioramento dell'errore a regime, in accordo con i risultati trovati nel paragrafo 3.2.2.

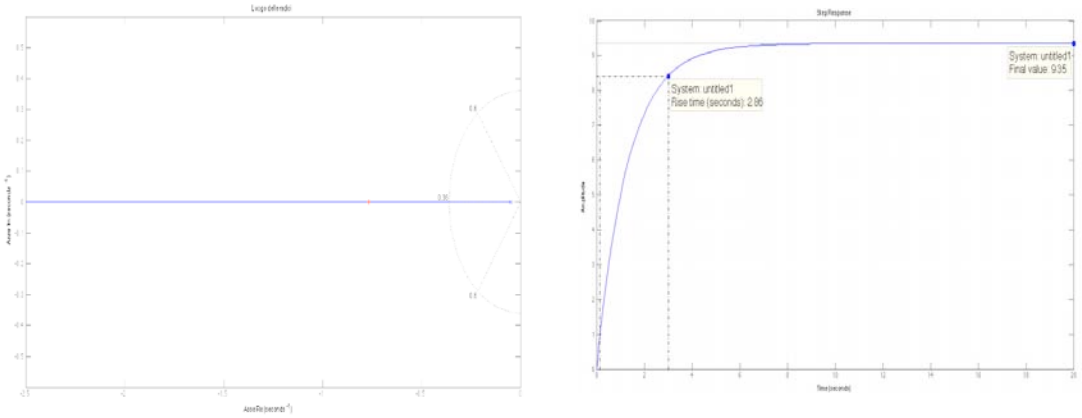


Figura 3.11: Luogo delle radici e corrispondente risposta al gradino
Punto scelto in -0.7672 + 0.0019i e rispettivo K_p=717.18

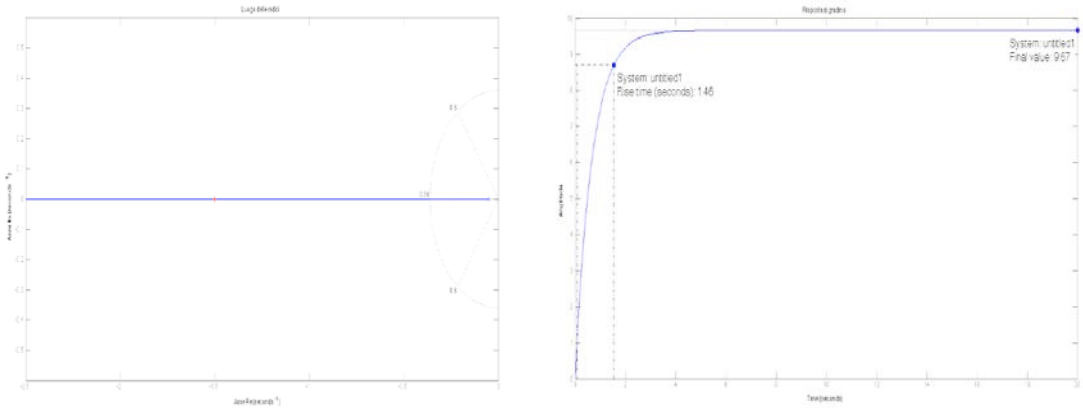


Figura 3.12: Luogo delle radici e corrispondente risposta al gradino
Punto scelto in -1.5018 + 0.0019i e rispettivo K_p=1451,8

3.3.3 Inserimento di una rete ritardatrice (*Lag controller*)

Nel paragrafo precedente abbiamo evidenziato come l'inserimento di una costante proporzionale K_p non riesca a risolvere sia il problema sul tempo di salita, sia quello sull'errore a regime. Un aumento incontrollato di K_p al fine di ridurre quest'ultimo darebbe luogo ad un *rise time* meccanicamente impossibile da ottenere.

Per ridurre l'errore a regime si ha quindi bisogno di aggiungere una rete ritardatrice (*lag controller*) al sistema retroazionato, la cui funzione di trasferimento è:

$$C_{lc}(s) = \frac{s + z_0}{s + p_0} \tag{3.13}$$

formata da una coppia zero (z_0) polo (p_0), posizionati in modo tale che la pulsazione del primo (ω_z) sia maggiore della pulsazione del secondo (ω_p).
Il diagramma di Bode dei moduli evidenzia che il sistema non attenua né amplifica il segnale di ingresso finché $\omega < \omega_p$, mentre introduce un'attenuazione di 20 decibel per ogni decade di aumento di ω quando $\omega_p < \omega < \omega_z$; infine, per $\omega > \omega_z$, non si registra più alcuna modificazione a causa dei contributi uguali ed opposti di polo e zero nel sistema.
Per quanto riguarda il diagramma delle fasi, si osserva che il sistema non opera alcuno sfasamento sul segnale di ingresso finché esso ha pulsazione $\omega < \omega_p$ o $\omega > \omega_z$. Per pulsazioni comprese nell'intervallo $\omega_p < \omega < \omega_z$, invece, il sistema introduce uno sfasamento negativo, cioè un *ritardo di fase* (da qui il nome assegnato alla rete).

Sostituendo la (3.13) nella (3.4) (escludendo il guadagno K_p) con la G(s) definita dalla (1.5), la funzione di trasferimento ad anello chiuso diventa:

$$H(s) = \frac{s + z_0}{Ms^2 + (b + Mp_0)s + bp_0} \tag{3.14}$$

Quindi, inserendo il guadagno K_p, la (3.14) diventa finalmente:

$$H(s) = \frac{K_p s + K_p z_0}{Ms^2 + \big(b + Mp_0 + K_p\big)s + (bp_0 + K_p z_0)} \tag{3.15}$$

Rispettando l'ordine descritto all'inizio di questo paragrafo e, per queste ragioni, scegliendo di collocare lo zero nel punto di pulsazione ω_z=0.3 rad/s e il polo in ω_p=0.03 rad/s, possiamo tracciare il luogo delle radici del sistema con i seguenti comandi *Matlab*:

```
M = 1000;        %massa del veicolo
b = 50;          %coefficiente complessivo di attrito
r = 10;          %velocità in m/s assunta costante
s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento G(s)
z0 = 0.3;        %scelta zero
p0 = 0.03;       %scelta polo
C_lc = (s + z0) / (s + p0);     %FdT rete ritardatrice
rlocus (C_lc * F_t);
axis ([-0.6 0 -0.4 0.4])
sgrid (0.6 , 0.36);
```

Il grafico ottenuto è riportato nella figura 3.13 sottostante:

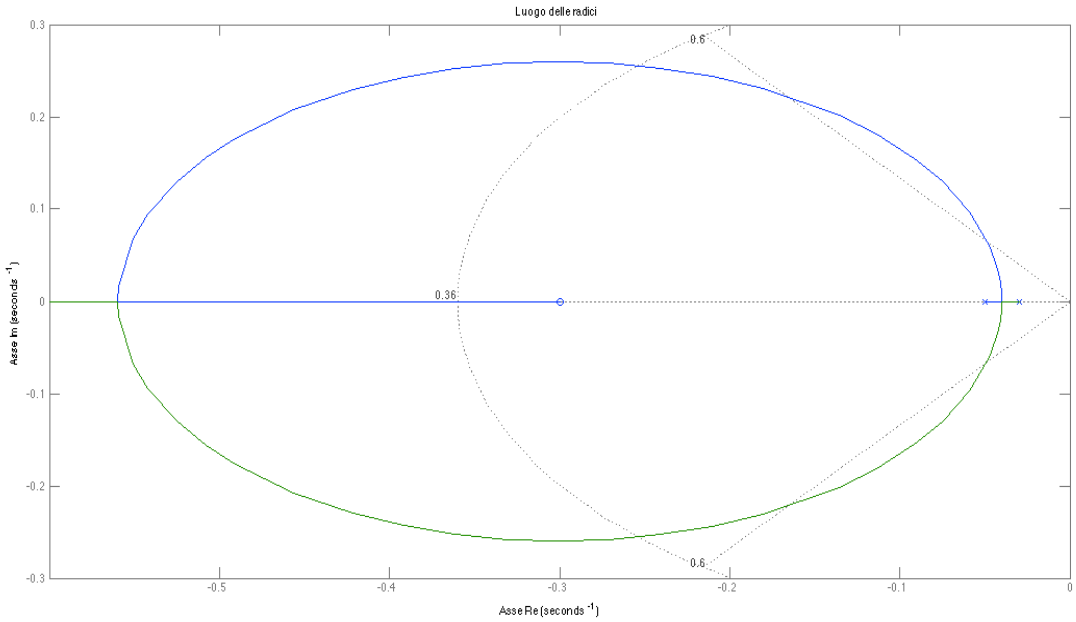


Figura 3.13: Luogo delle radici con rete ritardatrice

Come fatto prima, usando il comando "*rlocfind*" possiamo far variare il valore del guadagno K_p al fine di rispettare le specifiche di progetto. Con il comando:

```
[Kp, poles] =rlocfind (C_lc * F_t)
```

possiamo scegliere un punto sull'asse reale prossimo a -0.4 (la scelta del punto rispetta le considerazioni viste precedentemente), per il quale il risultato *Matlab* è:

```
Select a point in the graphics window

selected_point =
-0.4017 - 0.0012i
Kp =
1.2856e+03
poles =
-0.9639
-0.4017
```

Il guadagno è allora $K_p = 1.2856 * 10^3$.

Adesso possiamo infine generare la nuova risposta al gradino per il sistema ad anello chiuso come segue:

```
Kp = 1.2856e+03;          %Impone il valore per la il guadagno
Fb = feedback (Kp * C_lc * F_t, 1)     %collega più blocchi
                          %(FdT) in un sistema in retroazione
t = 0:0.1:20;
step (r * Fb, t)
axis([0 20 0 12])
```

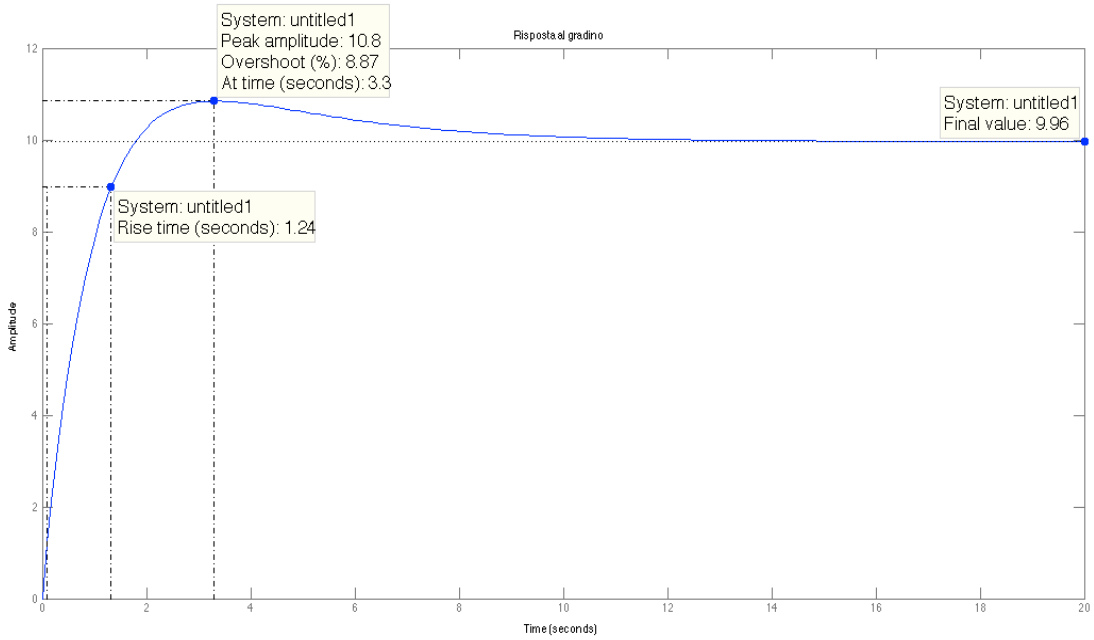


Figura 3.14: Risposta al gradino con K_p ottenuto automaticamente

Come si può osservare dalla risposta al gradino, l'errore a regime è stato ridotto quasi a zero (ma comunque entro i limiti richiesti), ed anche il tempo di salita rispetta le specifiche.

La sovraelongazione (dell'8.87%, quindi quasi a ridosso del limite progettuale ma comunque accettabile) è causata dall'inserimento di uno zero aggiuntivo nel sistema complessivo.
Nonostante tutti i criteri di progettazione siano stati rispettati, è comunque utile sperimentare diversi valori per il posizionamento del polo e/o dello zero, per verificare qual è il loro effetto sulla risposta al gradino del sistema retroazionato.

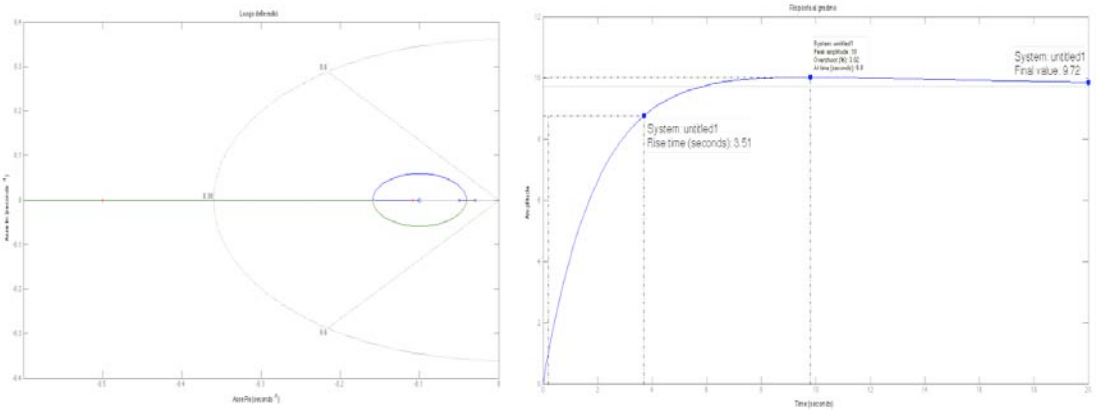


Figura 3.15: Luogo delle radici e risposta al gradino con z_0=0.1 e K_p=528.6507

Avevamo detto che lo zero aggiuntivo nel sistema era la causa della sovraelongazione, pertanto modificando la sua posizione siamo in grado di correggerne il valore. Nella figura 3.15, scegliendo z_0=0.1 e p_0=0.03, siamo riusciti ad avere una sovraelongazione inferiore rispetto al caso precedente (∼3 %). Inoltre la diminuzione del guadagno K_p comporta un aumento del tempo di salita e dell'errore a regime (siamo comunque ancora entro i margini di progetto).
Altre modificazioni dei parametri comportano l'ottenimento di risultati differenti.

3.3.4 Considerazioni finali

Abbiamo applicato il metodo del luogo delle radici per analizzare il comportamento del sistema ad anello chiuso, per osservarne l'ottemperanza ai vincoli di progetto. Poiché il metodo è basato sulla variazione di un parametro K, abbiamo ricavato le stesse conclusioni trovate quando è stato analizzato il controllore proporzionale **P**, cioè l'impossibilità di ottenere risultati soddisfacenti, in particolare per l'errore a regime, se non aumentando il valore di K, con il rischio però di incorrere in un'incontrollabilità fisica del sistema.
Abbiamo quindi aggiunto una rete ritardatrice, inserendo una coppia zero-polo, che ha permesso da un lato di migliorare l'errore a regime, dall'altro ha introdotto una

sovraelongazione non trascurabile. Scegliendo opportunamente i valori per le pulsazioni ω_p e ω_z e per il guadagno K_p, siamo comunque in grado di rispettare tutte le specifiche richieste dal problema, come esposto nel paragrafo 1.4.

3.4 Analisi in frequenza

3.4.1 Introduzione all'analisi in frequenza

Fino ad ora abbiamo sempre analizzato la nostra funzione di trasferimento nel dominio del tempo, verificando che essa potesse rispettare alcuni parametri come il tempo di salita, la sovraelongazione o l'errore a regime.
È possibile tuttavia anche effettuare un'analisi in frequenza, considerando cioè la funzione di trasferimento $W(j\omega)$, chiamata *risposta in frequenza*, ottenuta a partire da W(s) semplicemente sostituendo il parametro "s" del dominio di Laplace con $j\omega$.
Il metodo della risposta in frequenza può sembrare meno intuitivo rispetto a quelli visti in precedenza; tuttavia esso presenta numerosi vantaggi, specialmente nelle situazioni della vita reale, come per la modellizzazione di funzioni di trasferimento ottenute a partire da dati fisici raccolti sperimentalmente.
Per poter utilizzare questo procedimento, vediamo di introdurre alcuni parametri che fanno riferimento alla $W(j\omega)$, così come avevamo fatto all'inizio per l'analisi nel dominio del tempo:

- **Banda passante (a 3 dB) (B_p):** individua l'insieme delle pulsazioni nell'intervallo $[0, \omega_b]$, essendo ω_b la pulsazione alla quale il modulo del diagramma di Bode relativo alla risposta in frequenza vale -3 dB.

  $$B_p = \max\{\omega_b \geq 0 : \forall \omega \in [0, \omega_b], |W(j\omega)|_{dB} \geq |W(0)|_{dB} - 3\ dB\}$$

  dove $W(0)=K_b$, se W(s) è strettamente propria;

- **Pulsazione di risonanza (ω_r):** indica, se esiste, quell'unica pulsazione positiva tale che valga la seguente condizione:

  $$\forall \omega \geq 0, \omega \neq \omega_r\ , |W(j\omega)| < |W(j\omega_r)|$$

  E' quindi quella pulsazione positiva per la quale il modulo della risposta in frequenza assume il valore massimo (può non esistere se, ad esempio, il punto di massimo è in corrispondenza di una pulsazione non positiva);

- **Picco di risonanza relativo (M_{rel}):**

$$M_{rel} = |W(j\omega_r)|_{dB} - |W(0)|_{dB}$$

  Esiste se esiste la pulsazione di risonanza, che è quindi l'ascissa corrispondente al punto di ordinata M_{rel}.

Ci si chiede ora com'è possibile riportare sul piano bidimensionale $\mathbb{R}^2$ la rappresentazione grafica della risposta in frequenza $W(j\omega)$ che, essendo complessa, giace in $\mathbb{R}^3$.

I due diagrammi più comuni che sono utilizzati a tale scopo sono:

1. Il *diagramma di Bode*, che consiste in due grafici che rappresentano rispettivamente il modulo (o ampiezza) e la fase della risposta in frequenza. La loro rappresentazione è fatta su di un piano cartesiano avente in ascissa, come variabile indipendente, la frequenza o la pulsazione e in ordinata il modulo dell'ampiezza, usualmente espressa in decibel o il modulo della fase espressa in gradi o radianti;

2. Il *diagramma di Nyquist* che, al contrario, offre una rappresentazione della risposta in frequenza nel piano complesso, al variare della pulsazione.

Per concludere questa introduzione, vengono infine presentati altri tre parametri di fondamentale importanza:

- **Pulsazione di attraversamento (ω_a):** è il punto del diagramma di Bode in cui il guadagno vale 0 dB; in pratica, sono tutti i punti in cui il diagramma di Bode delle ampiezza interseca l'asse delle ascisse;

- **Fase di attraversamento (φ_a):** è la fase corrispondente alla pulsazione ω_a;

- **Margine di fase (m_φ):** è la grandezza $\varphi_a + \pi$, che corrisponde alla distanza angolare sul diagramma di Nyquist dal punto -1. Nel diagramma di Bode, esso può essere ricavato calcolando, in corrispondenza della pulsazione di attraversamento, la differenza tra la fase effettiva e la fase -180°.

3.4.2 Diagramma di Bode e risposta in catena aperta

Il primo passo per risolvere il problema utilizzando la risposta in frequenza è determinare la funzione di trasferimento ad anello aperto da usare.
Proprio come per il metodo del luogo delle radici, ci si servirà unicamente di un controllore proporzionale di tipo P. Il diagramma a blocchi e la funzione di trasferimento in catena aperta sono riportati nella figura sottostante:

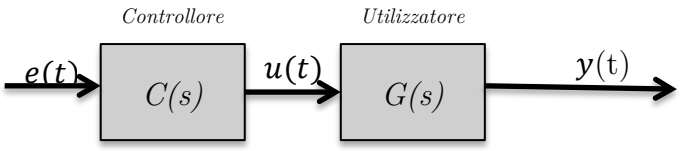


Figura 3.17: Diagramma a blocchi della funzione di trasferimento

$$\frac{Y(s)}{E(s)} = \frac{K_p}{Ms + b} \tag{3.16}$$

Verifichiamo nuovamente che la risposta ad anello aperto é stabile. Consideriamo pertanto K_p=1 ed osserviamo, con i seguenti comandi *Matlab*, come risulta la risposta al gradino del sistema ad anello aperto:

```
M = 1000;   %massa del veicolo
b = 50;     %coefficiente complessivo di attrito
u = 500;    %forza impressa al sistema
Kp = 1;
s = tf ( 's' );
F_t = 1 / (M * s + b);  %funzione di trasferimento
C = Kp;     %assegna al C(s) la sola componente proporzionale
step (u * C * F_t)
```

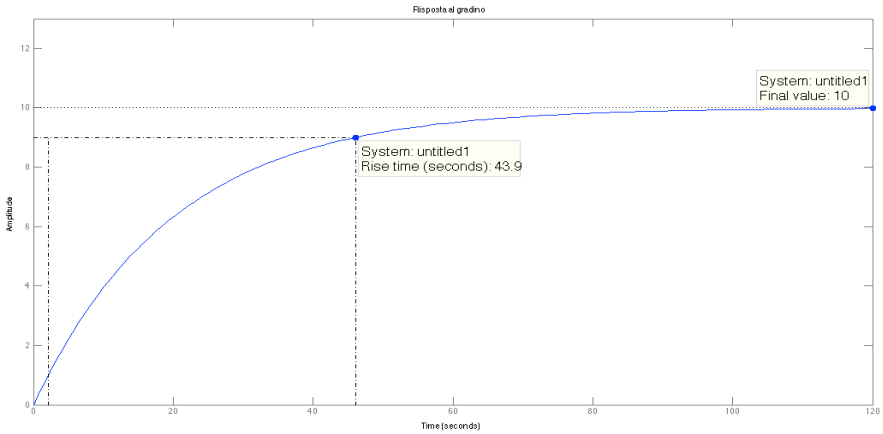


Figura 3.18: Risposta al gradino con K_p=1

È possibile osservare che effettivamente il sistema a ciclo aperto è stabile. Passiamo adesso all'analisi vera e propria, generarando il diagramma di Bode. È necessario fornire solamente il comando `bode (C * F_t)`;

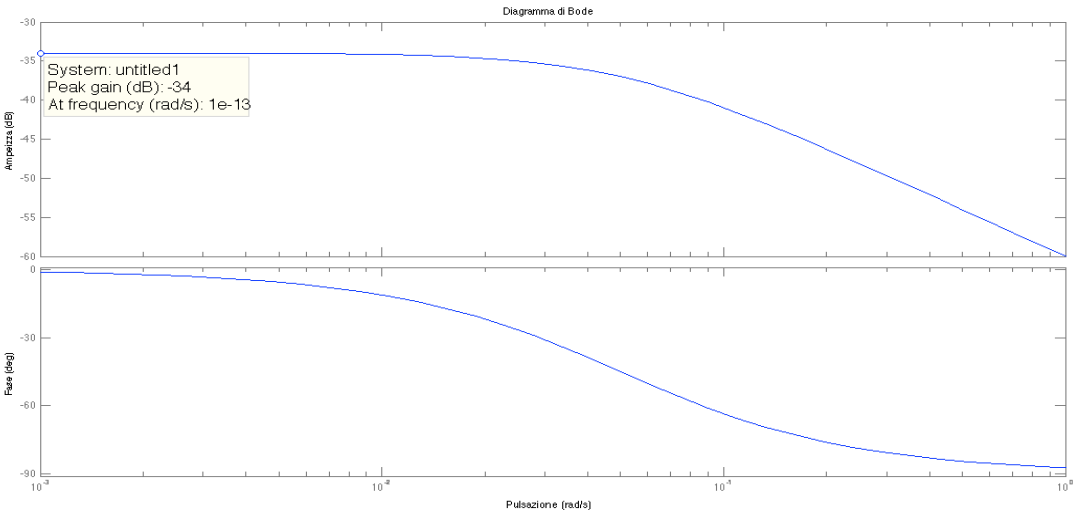


Figura 3.19: Diagramma di Bode di modulo e fase

3.4.3 Analisi della risposta in catena chiusa

Vediamo di osservare cosa è possibile dedurre dal diagramma di Bode riportato nella figura 3.19. L'errore a regime permanente (*steady-state error*) può essere ricavato dalla seguente equazione:

$$ss_{error} = \frac{1}{1 + G_{\omega \to 0}} \cdot 100\% \tag{3.17}$$

dove $G_{\omega \to 0}$ è il guadagno a bassa frequenza che vale -34 dB. In scala lineare, esso vale:

$$G = 10^{\frac{G_{\omega \to 0}[dB]}{20}} \cong 0.02 \tag{3.18}$$

Pertanto l'errore a regime vale $ss_{error} = \frac{1}{1+0.02} \cdot 100\% \cong 98\%$, notevolmente sballato rispetto ai vincoli progettuali. Possiamo confermare ciò generando la risposta al gradino del sistema ad anello chiuso con i seguenti comandi aggiuntivi:

```
r = 10;     %velocità in m/s assunta costante (è l'ingresso)
Fb = feedback (C * F_t, 1); %collega più blocchi (FdT) in un
                            %unico sistema in retroazione
step (r * Fb);
```

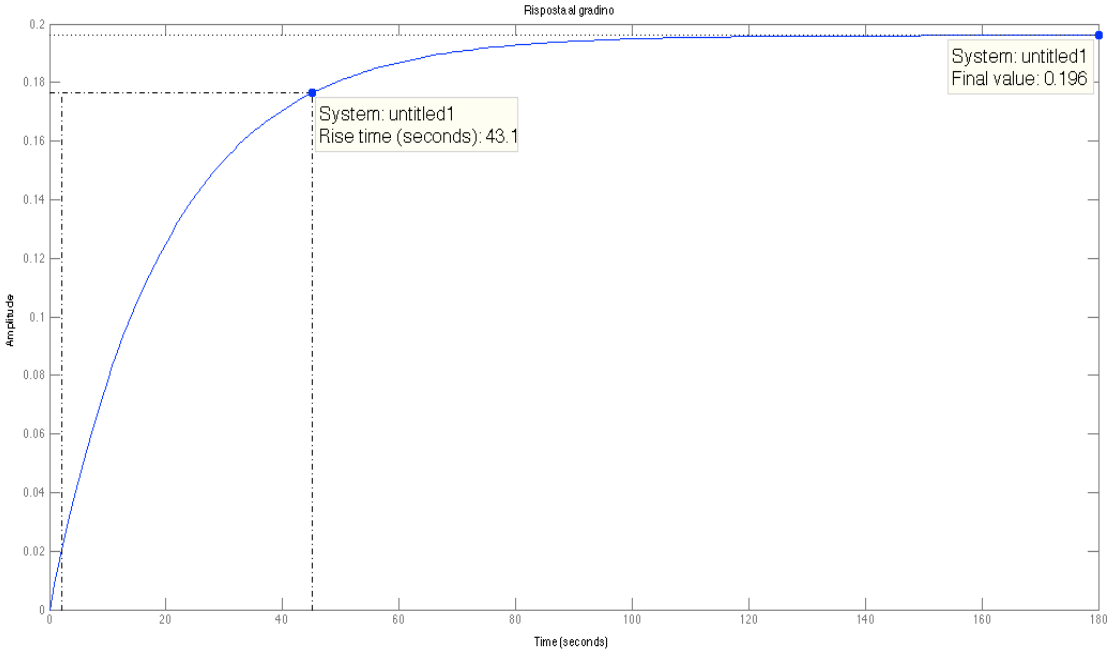


Figura 3.20: Risposta al gradino del sistema a catena chiusa con guadagno unitario

Per migliorare l'errore a regime e riportarlo entro i limiti richiesti, è necessario aumentare il guadagno a bassa frequenza, in modo da aumentare il denominatore dell'equazione (3.17) e quindi ridurre l'ss_{error}..
In particolare, l'errore deve essere <2% e pertanto devono essere rispettate le seguenti condizioni:

$$ss_{error} < 2\% \Rightarrow \frac{1}{1 + G_{\omega \to 0}} \cdot 100\% < 0.02 \Rightarrow G_{\omega \to 0} > 49 = 33.8\ dB$$

Per poter raggiungere l'errore a regime desiderato, è necessario far compiere una traslazione verso l'alto del diagramma di Bode di almeno 33.8 – (-34) = 67.8 dB, cioè di un valore linearizzato di circa 2455. Assegniamo quindi al parametro K_p il valore appena trovato, il quale dovrebbe garantire di ottenere un ss_{error} entro le specifiche.

Il diagramma di Bode risultante è il seguente:

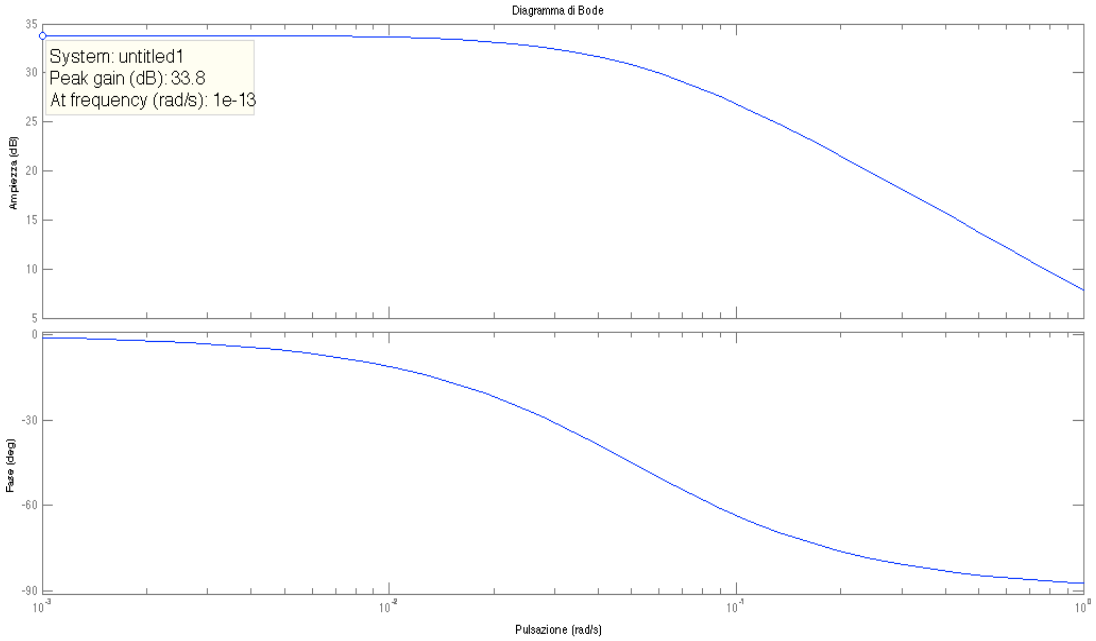


Figura 3.21: Diagramma di Bode di modulo e fase con K_p=2455

Come si può notare dalla figura 3.21, adesso il guadagno a bassa frequenza è di 33.8 dB, come richiedevamo. Pertanto la risposta al gradino del sistema ad anello chiuso dovrebbe rispettare, per quanto concerne l'errore a regime permanente, le richieste di progetto. Verifichiamolo, utilizzando gli oramai noti comandi *Matlab*.

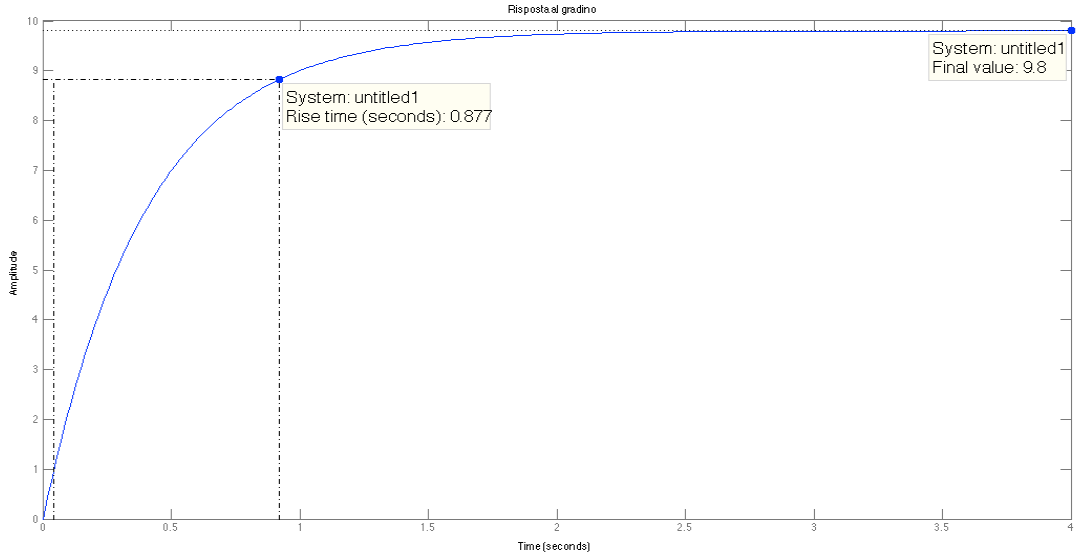


Figura 3.22: Risposta al gradino del sistema a catena chiusa con K_p=2455

Possiamo in effetti constatare che l'errore a regime soddisfa i requisiti. Tuttavia il tempo di salita è fin troppo basso e alquanto irragionevole, dal momento che un'automobile non può accelerare da 0 a 9 m/s in meno di un secondo. Questo è ancora una volta in accordo con i risultati trovati in tutte le analisi precedenti, nelle quali si osservava che un aumento eccessivo del guadagno K_p avrebbe inevitabilmente portato il sistema in una condizione di assoluta irrealizzabilità fisica.
Pertanto da un lato dovremo utilizzare un guadagno proporzionale più piccolo, per aumentare il tempo di salita a costo di un aumento dell'errore a regime, dall'altro sarà richiesta la progettazione di una rete ritardatrice per ridurre tale errore.

3.4.4 Inserimento di una rete ritardatrice (*Lag controller*)

Avevamo già introdotto la rete ritardatrice ancora nel paragrafo 3.3.3, nel quale abbiamo sottolineato come essa sia necessaria per migliorare l'errore a regime, anche a costo di dover gestire una sovraelongazione non trascurabile (che deve essere mantenuta entro i limiti progettuali). In aggiunta, riducendo il guadagno proporzionale, siamo anche il grado di riportare il *rise time* entro dei valori accettabili. Il *lag controller* è pertanto ciò di cui abbiamo bisogno.
Ricordiamo la (3.13), che esprime la funzione di trasferimento per una rete ritardatrice:

$$C_{lc} = \frac{s + z_0}{s + p_0}$$

E' noto inoltre che, con l'uso di una rete ritardatrice, si riesce a ridurre l'errore a regime di un fattore z_0/p_0. Assegnando allora allo zero il valore z_0=0.1 e al polo il valore p_0=0.02, riusciamo a ridurre l'ss_{error} di un fattore $\frac{z_0}{p_0} = 5$. Scegliamo infine per il guadagno proporzionale il valore K_p=600 (dalle analisi precedenti, infatti, con una simile costante riusciamo ad ottenere un tempo di salita ragionevole).
Di seguito sono riportati i comandi *Matlab* che useremo per generare il diagramma di Bode e la risposta al gradino del sistema complessivo ad anello chiuso:

```
Kp = 600;
zo = 0.1;       %scelta zero
po = 0.02;       %scelta polo
C_lc = (s+zo)/(s+po);   %FdT della rete ritardatrice
bode(Kp*C_lc*F_t);
```

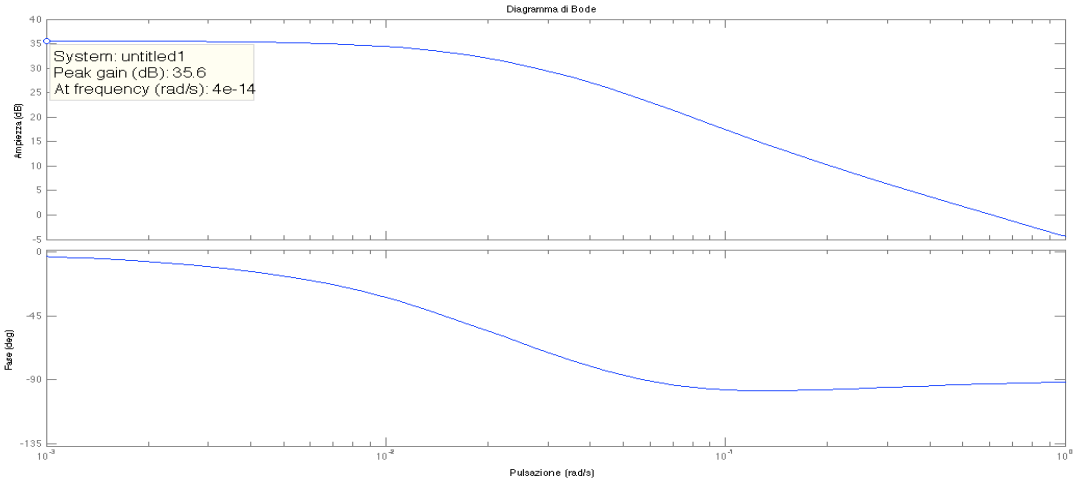


Figura 3.23: Diagramma di Bode di modulo e fase con K_p=600 e rete ritardatrice

La conferma che i parametri di progetto sono rispettati arriva con l'analisi del grafico della risposta al gradino del sistema a catena chiusa:

```
Fb = feedback(Kp * C_lc * F_t,1);
t = 0:0.1:20;
step(r*Fb,t);
```

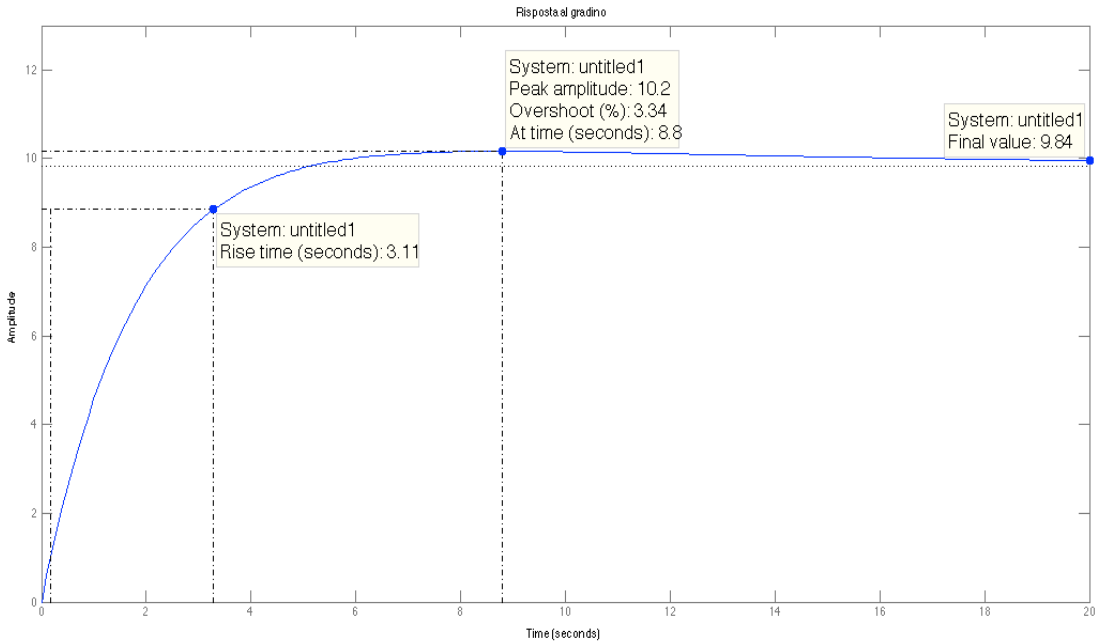


Figura 3.24: Risposta al gradino (sistema anello chiuso) con K_p=600 e rete

Tutte le richieste sui parametri sono rispettate:

- Il *rise time* è di 3.11 secondi ($<$ 5 secondi richiesti);
- La sovraelongazione (*overshoot*) è del 3.34 % ($<$ 10% richiesto);
- L'errore a regime permanente è 9.84, cioè pari al $\left(1 - \frac{9.84}{10}\right) \cdot 100\% = 1.6\%$ ($<$ 2% richiesto).

3.4.5 Considerazioni finali

Anche in quest'analisi, abbiamo riscontrato le stesse conclusioni dei paragrafi precedenti. Utilizzando solo un controllore proporzionale riusciamo a rispettare tutte le specifiche di progetto, ma andiamo incontro a risultati inaccettabili dal punto di vista meccanico sul tempo di salita. Inserendo anche una rete ritardatrice, invece, con la scelta di opportuni parametri, riusciamo a raggiungere tutti gli obbiettivi preposti su *rise time*, *overshoot* e *steady-state error*.

CAPITOLO 4
Modellizzazione del progetto con *Simulink*

4.1 Introduzione a *Simulink*

Simulink è un software per la modellazione, simulazione e analisi di sistemi dinamici, sviluppato dalla compagnia statunitense *MathWorks* al fine di renderlo strettamente integrato con l'applicazione *Matlab*.
Tramite questo strumento, è molto semplice rappresentare e poi simulare un modello matematico che rappresenta un sistema fisico. Tali modelli sono riprodotti come diagrammi schematici; *Simulink* possiede un'estesa libreria virtuale, nella quale è possibile individuare una vasta gamma di blocchi da impiegare per la costruzione del sistema.
Uno dei vantaggi principali per i quali si utilizza questo software per l'analisi dei sistemi dinamici è che esso permette di analizzare rapidamente la risposta di sistemi complicati, operazione che può essere alquanto proibitiva e difficile da calcolare per via analitica.

Attraverso un'equazione differenziale matematica che descrive il sistema, è quindi possibile creare lo schema a blocchi per il sistema stesso, quindi simularne automaticamente il funzionamento.
Per l'analisi del *cruise control* che stiamo conducendo, avevamo individuato nel capitolo 1 l'equazione differenziale (1.1) che rappresenta analiticamente in sistema, illustrato dalla figura 1.1:

$$M\dot{v} = u - bv \tag{4.1}$$

dove "u" è la forza impressa dal veicolo, che può essere direttamente controllata.
Manteniamo invariati anche i valori di riferimento degli altri parametri di progetto, cioè:

- Massa (M) = 1000 kg
- Coefficiente complessivo di attrito (b) = 50 Ns/m
- Forza impressa dal veicolo (u) = 500 N
- Velocità di riferimento (r) = 10 m/s

4.2 Costruzione del modello

Questo sistema sarà modellato sommando le forze che agiscono sulla massa e integrando l'accelerazione per ottenere la velocità del veicolo.
Per prima cosa, iniziamo modellando l'integrale di accelerazione, a partire dalla (4.1), con l'inserimento di un opportuno blocco:

$$\dot{v} = \frac{u(t) - bv}{M} \quad \Longrightarrow \quad v = \int \frac{u(t) - bv}{M}\, dt \tag{4.2}$$

Aprendo una nuova finestra dal programma *Simulink*, eseguiamo le seguenti operazioni:

1. Dalla libreria "*continuous*", inserire un blocco integratore e disegnare le linee da e verso i terminali di ingresso e di uscita, etichettando la linea di ingresso come "accelerazione" e la linea d'uscita come velocità.

Lo schema che otteniamo è rappresentato nella figura 4.1:

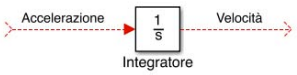


Figura 4.1: Schema a blocchi dell'integratore

La freccia che rappresenta l'accelerazione (dv/dt) è uguale alla somma algebrica delle forze agenti, divise per la massa: $\frac{dv}{dt} = \frac{u(t)-bv(t)}{M}$.

Possiamo allora completare lo schema a blocchi nel modo che segue:

2. Dalla libreria "*math operations*", inserire un blocco guadagno (chiamato "massa" e del valore 1/M) collegato alla linea di ingresso del blocco integratore e disegnare una linea che porta all'ingresso del blocco appena creato;

3. Collegare un blocco sommatore alla linea di ingresso del blocco massa, modificando uno dei due segni da + a - . Esso serve per effettuare l'operazione di differenza tra ut(t) e bv(t);

4. Inserire un ulteriore blocco guadagno (chiamato "attrito" e del valore b), la cui uscita è collegata al segno meno del blocco sommatore e il cui ingresso è collegato all'uscita dell'integratore;

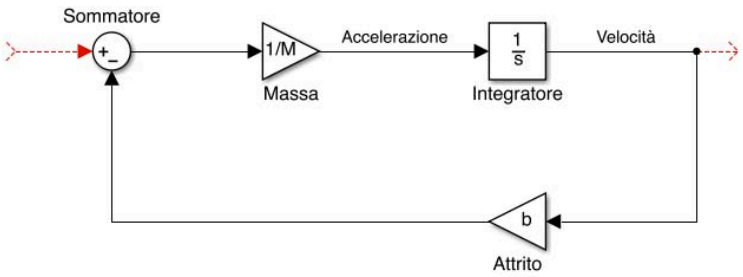


Figura 4.2: Schema a blocchi in costruzione

5. Applicare al morsetto positivo del blocco sommatore l'ingresso del sistema. Scegliamo un ingresso a gradino, inserendo un blocco gradino dalla libreria "*souces*";

6. Per visualizzare il valore della velocità in uscita, inserire un blocco di visualizzazione dalla libreria "*sinks*", collegato alla freccia d'uscita dell'integratore.

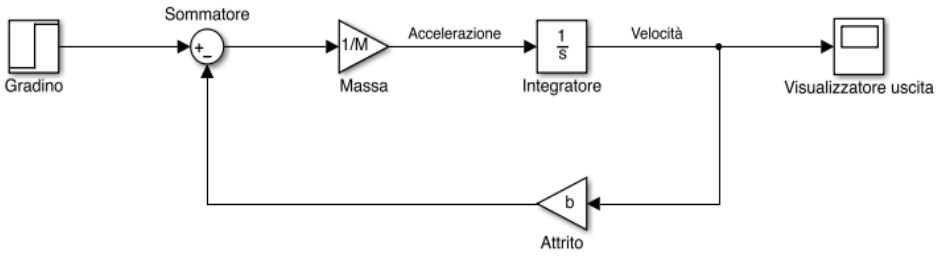


Figura 4.3: Schema a blocchi finale

Come ultimo passaggio, per fornire un valore d'ingresso adeguato, è necessario impostare lo *step time* a "0" e il valore finale a "u"; serve quindi un tempo di stop di

almeno 120 secondi [8]. Quest'ultimo valore può essere impostato dal comando *Parateters* nel menu *Simulations.*

4.2.1 Risposta del sistema in catena aperta

Dalla finestra di prompt dei comandi di *Matlab*, si inseriscono i valori per i principali parametri fisici (massa, coefficiente di attrito e forza impressa dal veicolo), che saranno utilizzati nella simulazione.

Per eseguire la simulazione, basta selezionare il comando start dal menu *Simulation.* Quando essa è terminata, basterà fare doppio clic sul blocco per la visualizzazione dell'uscita per osservare il seguente output:

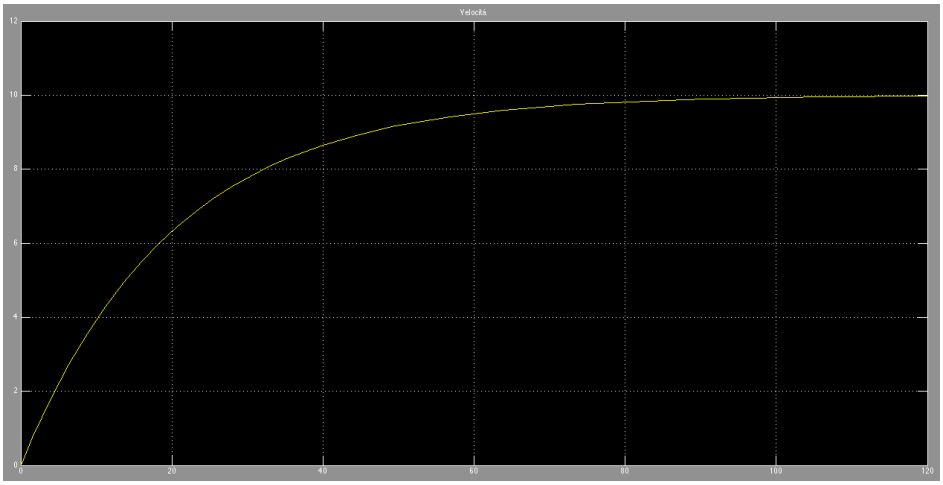


Figura 4.4: Risposta del sistema simulato in catena aperta

La simulazione riportata nella figura 4.4 ci espone i medesimi risultati trovati nelle analisi precedenti. Pertanto, in assenza di sistemi di controllo, il sistema non rispetta le specifiche di progetto, in particolare sul tempo di salita (*rise time*), che risulta essere di molto superiore ai 5 secondi richiesti. Al contrario, sono rispettati i valori di sovraelongazione ed errore a regime.

[8]Questo tempo è più che sufficiente per osservare la risposta del sistema ad anello aperto completa, come abbiamo avuto modo di constatare dalle numerose analisi svolte nei capitoli precedenti.

C'è allora la necessità di modellare un opportuno sistema di controllo, al fine di rispettare i parametri imposti.

4.3 Modellizzazione di un sistema di controllo

4.3.1 Estrazione di un modello lineare in *Matlab*

La figura 4.3 rappresenta la schematizzazione completa per il sistema in catena aperta che stiamo analizzando. Per agevolare l'analisi successiva, ci proponiamo di racchiudere tutti i blocchi presenti in un unico "sottosistema" che, una volta impiegato, possa rappresentare equivalentemente l'insieme dei blocchi, così come riprodotti nella figura 4.3.
Per fare ciò, basta sostituire il blocco "gradino" con "In1" e il blocco "visualizzatore uscita" con "Out1". Il risultato di quest'operazione è rappresentato nella figura sottostante:

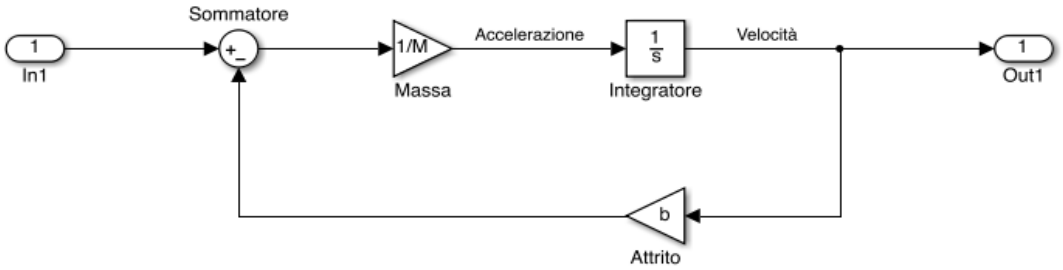
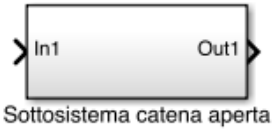


Figura 4.4: Schema a blocchi, che sarà rappresentato da un unico sottosistema

Dopo aver svolto queste semplici operazioni, sarà possibile inserire un unico sottosistema in sostituzione di tutti i blocchi che si trovano nella figura 4.3; il funzionamento è ovviamente equivalente, e in più questo espediente consente di rendere più agevole la schematizzazione che andremo a fare.

Inseriamo allora, in un nuovo file *Simulink*, dalla libreria "*ports & subsistems*", un blocco "sottosistema" e, una volta aperto con un doppio clic, incolliamo al suo interno tutto ciò che è rappresentato nello schema 4.4.



Adesso e ogni volta che sarà impiegato, il blocco “sottosistema catena aperta” sostituirà in tutto e per tutto l’insieme di blocchi memorizzati al suo interno.

4.3.2 Implementazione di un controllore PI

Nei capitoli precedenti abbiamo visto che l’uso di un controllore proporzionale-integrale di tipo PI (con la scelta di opportuni parametri, come ad esempio K_p=1000 e K_i=50[9]) consente di rispettare tutte le specifiche di progetto che ci sono imposte.
Come prima cosa bisogna allora modellizzare il sistema a catena chiusa retroazionato inserendo, oltre al blocco che rappresenta il sottosistema, anche dei blocchi aggiuntivi per la funzione di trasferimento che implementa il compensatore e per l’ingresso e l’uscita. Seguiamo i seguenti passaggi:

1. Inserire un blocco sommatore con segni “ + - ”; l’uscita del sottosistema in catena aperta è collegato con il segno “-” del sommatore;

2. Inserire un blocco integratore dopo il blocco sommatore. L’uscita di quest’ultimo blocco fornirà il segnale di errore;

3. Inserire un blocco guadagno (chiamato “guadagno integrale” e del valore K_i) collegato all’uscita del blocco integratore;

4. Inserire un ulteriore blocco guadagno (chiamato “guadagno proporzionale” e del valore K_p) collegato anch’esso all’uscita del blocco integratore;

Dopo aver definito le costanti integrali e proporzionali, possiamo sommarle per ricavare la funzione di trasferimento del controllore PI, come espresso dalla (3.7). Infatti, sommando $\frac{1}{s}K_i$ e K_p otteniamo proprio la (3.7).

5. Inserire un blocco sommatore, cui sono collegate le uscite di entrambi i blocchi di guadagno presenti nel modello, e collegare la sua uscita con l’ingresso del sottosistema catena aperta;

[9] Questi valori per i parametri di controllo non sono scelti arbitrariamente. Ancora nel paragrafo 3.2.3 le analisi fatte avevano sottolineato come tali parametri permettessero di rispettare pienamente le imposizioni progettuali.

Infine, applichiamo un ingresso a gradino collegato al morsetto positivo del primo sommatore e un blocco per la visualizzazione dell'uscita, direttamente collegato all'uscita del sottosistema. Bisogna anche ricordarsi di impostare, per il gradino d'ingresso, lo *step time* a "0" e il valore finale a "u".

Complessivamente, il modello ottenuto è illustrato nella figura sottostante[10]:

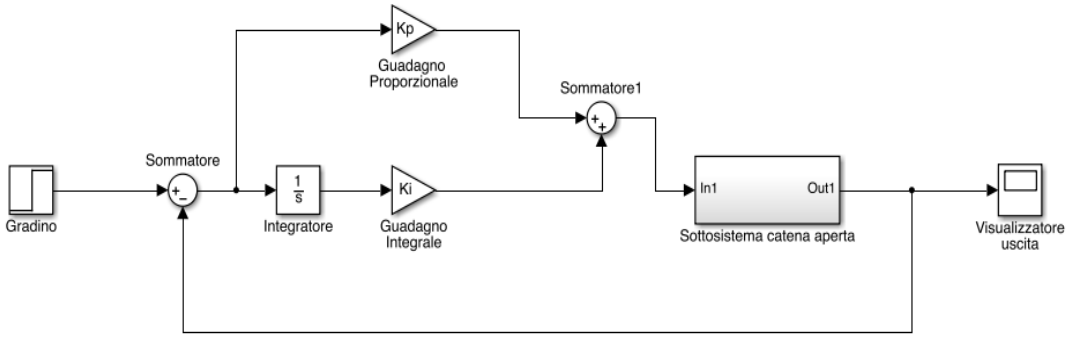


Figura 4.5: Schema a blocchi del sistema in catena chiusa

Nella figura 4.5 abbiamo costruito un compensatore PI a partire da blocchi fondamentali, sommati assieme. Tuttavia *Simulink* mette a disposizione, nella libreria *Continuous*, un blocco "funzione di trasferimento" per implementare il controllore in un unico passaggio.

Basterà allora fornire il valore della funzione di trasferimento del PI, così come espressa nella (3.7): $C_{PI} = \frac{K_p s + K_i}{s}$. Il risultato è raffigurato qua sotto:

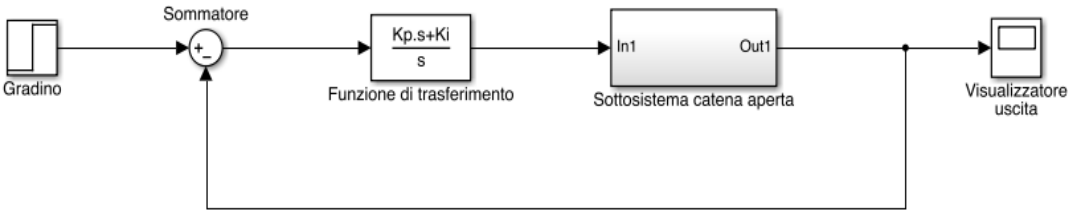


Figura 4.5 (bis): Schema a blocchi del sistema in catena chiusa

[10] Ricordiamo che il blocco "sottosistema catena aperta" racchiude equivalentemente l'insieme dei blocchi della figura 4.3, che rappresentano il sistema ad anello aperto.

4.3.3 Risposta del sistema in catena chiusa

Prima di passare alla simulazione vera e propria,impostiamo un tempo di stop di 10 secondi[11].
Ovviamente dobbiamo anche impostare, dal prompt dei comandi di *Matlab*, i valori dei parametri fisici (M=1000, b=50, u=10) e quelli del controllore (K_p=1000 e K_i=50). Facciamo quindi partire la simulazione e ciò che otteniamo dal visualizzatore d'uscita è rappresentato nella figura 4.6

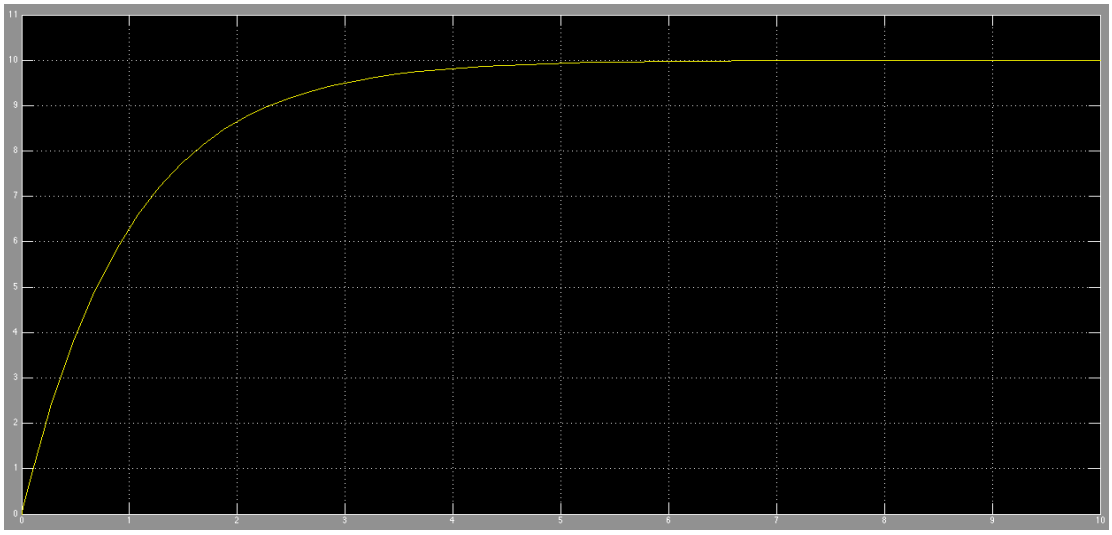


Figura 4.6: Risposta del sistema simulato in catena chiusa con controllore PI

Possiamo vedere che, solo con l'inserimento di un compensatore proporzionale-integrale, rispettiamo tutte le richieste di progetto: non c'è sovraelongazione, l'errore a regime è praticamente nullo e il tempo di salita è di circa 4 secondi (non solo è inferiore ai 5 s richiesti, ma è anche un valore fisicamente e meccanicamente ragionevole).

Come affermato nel capitolo 3.2.5, l'uso di un controllore PID renderebbe il sistema più complesso da realizzare e non gioverebbe nel miglioramento delle prestazioni, dal momento

[11] Dal momento che, se le specifiche sono rispettate, il sistema avrà un *rise time* massimo di 5 secondi, una simulazione su 10 secondi permetterà di visualizzare interamente la risposta del sistema.

che il sistema retroazionato con un compensatore proporzionale-integrale riesce a trovare da solo una risposta più che soddisfacente alle richieste progettuali, come abbiamo più volte avuto modo di notare.
L'analisi *Simulink* di un sistema con controllore PID è pertanto omessa in questa tesi.

CAPITOLO 5
Conclusioni

In questo elaborato si è discusso sulle migliori e più efficienti tecniche di modellizzazione per un sistema di cruise control, capace di regolare automaticamente la velocità di crociera di un'autovettura.
Dopo l'introduzione, necessaria per ricavare matematicamente le equazioni del modello che abbiamo impiegato in tutte le analisi successive, sono state presentate le diverse tecniche con cui si può modellizzare il sistema:

- Senza l'ausilio di alcun controllo, quindi considerando il sistema in "catena aperta";
- Impiegando controllori di tipo P, PI, PID;
- Impiegando una rete ritardatrice, sulla base dei risultati ottenuti con il metodo del luogo delle radici;
- Impiegando una rete ritardatrice, sulla base dei risultati ottenuti con un'analisi in frequenza.

Al termine di queste analisi, i risultati teorici sono stati provati da delle simulazioni reali, utilizzando il programma *Simulink*.
Ricordiamo inoltre che un modello corretto deve avere delle prestazioni entro tali limiti:

- Tempo di salita (Rise Time) < 5 s
- Sovraelongazione (Overshoot) $< 10\%$
- Errore a regime (Steady-State Error) $< 2\%$

Possiamo pertanto riassumere i risultati trovati nei diversi passaggi nella tabella sottostante, al fine di dedurre le conclusioni finali per questa tesi:

Controllore	Rise Time (s)	Overshoot	Errore regime
Nessuno	43.9	Nessuna	0
P	0.437	Nessuna	0.01 m/s
PI	2.2	Nessuna	Nessuno
PID	2.28	Nessuna	Nessuno
LAG Luogo radici	1.24	8.87%	0.04 m/s
LAG An. frequenza	3.11	3.34%	0.16 m/s

(Le caselle colorate di rosso identificano un risultato inaccettabile, quelle colorate di giallo rappresentano un risultato non ottimale ma comunque entro i limiti).

- L'analisi in catena aperta ha portato risultati altamente insoddisfacenti nel tempo di salita, sottolineando la necessità di fornire il progetto di un adeguato sistema di controllo;
- L'uso di un solo controllore proporzionale è stato considerato non apprezzabile per ragioni fisico-meccaniche, a causa dell'irrealistica rapidità con cui l'automezzo sarebbe stato in grado di raggiungere la sua velocità di regime[12];
- L'uso di reti ritardatrici (LAG) consente di rispettare tutte le specifiche di progetto, pur introducendo una sovraelongazione non trascurabile, a causa della presenza di uno zero nel sistema;
- L'uso di controllori di tipo PI e PID consente invece di rispettare a pieno e notevolmente entro i limiti tutte le richieste sulle prestazioni complessive.

Possiamo allora concludere che la progettazione di un sistema di cruise control richiede la presenza di un meccanismo di controllo, preferibilmente realizzato con compensatori PI o PID (generalmente il primo preferibile al secondo per la sua maggiore semplicità di realizzazione, non includendo la componente differenziale).

[12] Proprio per questo motivo sono state ritenuti insoddisfacenti tutte quelle altre modellizzazioni (sulla base delle analisi in frequenza o con il metodo del luogo delle radici) in cui si facesse ricorso al solo controllore P.

BIBLIOGRAFIA

[1]: Mauro Bisiacco, M. Elena Valcher, "*Controlli automatici*", Libreria Progetto Padova.

[2]: Control Tutorials Matlab&Simulink, Cruise Control Section; http://ctms.engin.umich.edu/CTMS/index.php?aux=Home

[3]: Mauro Bisiacco, Simonetta Braghetto, "*Teoria dei sistemi dinamici*", Progetto Leonardo, Esculapio, Bologna, ed. settembre 2011.

[4]: Cruise control informations and details:
http://ww2.autoscout24.it/glossario/cruise-control/181117/