VAB - Veicolo auto bilanciato

Relazione di progetto

Laboratorio di Sistemi Meccatronici II
Università degli Studi di Bergamo
Kilometro rosso
A.A. 2019/2020

Calegari Andrea - 1041183
Piffari Michele - 1040658

May 27, 2020

Contents

List of Figures

1

Introduzione

L'approccio seguito per la stesura del modello dinamico del veicolo autobilanciato ha da subito preso una via meno *tradizionale* rispetto al classico metodo risolutivo: abbiamo infatti preferito, data il nostro *background* informatico, approcciare il problema direttamente in ambiente Matlab, sfruttando sin da subito le potenzialità di calcolo offerte dal software di *Mathworks*.

Nello specifico, per la parte di stesura e definizione della dinamica, abbiamo inzialmente seguito una via risolutiva duale, portando avanti sia un'analisi letterale, sfruttando le potenzialità del **calcolo simbolico** messe a disposizione delle funzionalità di **live scripting**, sia uno studio numerico (considerando quindi le varie grandezze fisiche con i valori definiti delle specifiche di progetto).

In linea di massima lo sviluppo del progetto ha seguito un andamento a step graduali, cadenziati da incontri settimanali in cui poter confrontare e consolidare lo *stato di avanzamento dei lavori*: nello specifico, il lavoro ha seguito uno sviluppo in questa direzione step by step, rappresentabile in linea di massima da queste *pietre miliari*:

- **Dinamica di ogni singolo corpo rigido**: abbiamo impostato il problema della dinamica andando a considerare il veicolo auto bilanciato come un insieme di corpi rigidi di cui poterne studiare la dinamica in maniera separata;
- **Dinamica completa del VAB**: siamo andati poi a considerare il sistema nella sua completezza, andando ad unire i contributi dei corpi rigidi considerati in prima battuta singolarmente.

  Questo ci ha permesso di ottenere le equazioni del moto, in forma non lineare, le quali hanno permesso una completa descrizione del sistema che abbiamo modelizzato;
- **Linearizzazione**: siamo poi andati a linearizzare queste equazioni dinamiche (appunto non lineari) nell'intorno dell'equilibrio;
- **Definizione del controllo**: definizio del controllore tramite tecnica di *pole placement*;
- **Modellizzazione del motore**: introduzione del modello del motore sulla base delle caratteristiche contenute nella specifica di progetto;
- **Discretizzazione**: passaggio a tempo discreto per i segnali derivanti dal mondo analogico, ovvero per tutto ciò che concerne la parte di sensoristica;
- **Non idealità**: siamo andati a modellizzare anche la presenza di disturbi, di natura stocastica e dovuti alla quantizzazione;
- **Trasformazioni di blocchi in *interpreted function***: conversione delle funzionalità di controllo del motore e filtraggio dei segnali provenienti dai sensori in blocchi di codice *Matlab* per favorirne poi la successiva conversione ed implementazione a bordo del controllore;

TODO: note varie

Part I

Dinamica

2

Scomposizione del VAB

2.1 Considerazioni iniziali

Per il calcolo delle equazioni dinamiche del sistema siamo andati a considerare ogni singolo corpo rigido componente il sistema, calcolandone le grandezze fisiche di posizione e velocità, seguendo un approccio cartesiano. Nello specifico abbiamo considerato il sistema composto da:

- Asta
- Utente a bordo dello chassis
- Chassis (nel corso della trattazione sarà chiamata talvolta anche base)
- Ruota (che poi sarà considerata con un contributo, essendo il VAB composto da due ruote)

Ognuno di questi corpi rigidi separati è individuato da un punto, che ne rappresenta il centro di massa (o baricentro del corpo stesso): avremo quindi questo insieme di punti caratterizzanti il sistema (figura 2.1)

- P_a
- P_b
- P_c
- P_r

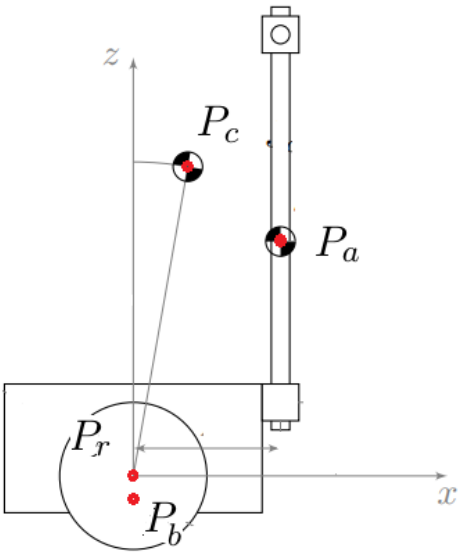

Figure 2.1: Baricentri dei singoli corpi rigidi

2.2 Grandezze di supporto

Prima di andare a definire le componenti di energia potenziale e cinetica di ogni singolo corpo, siamo andati ad introdurre alcune grandezze geometriche di supporto che definiremo qui di seguito.

Nello specifico abbiamo introdotto i seguenti parametri, specificati anche in figura 2.2:

- l_a: rappresenta la congiungente tra il centro del sistema di riferimento e il centro dell'asta, utilizzata appunto come manubrio, che abbiamo individuato come

$$\sqrt{(\frac{h_a}{2} + \frac{h_b}{2})^2 + (\frac{w_b}{2})^2}$$

- l_c: (TODO: check) questa grandezza invece rappresenta per noi l'altezza del baricentro del corpo dell'utente, la quale ovviamente andrà a dipendere dal valore di inclinazione del corpo stesso. Considerando il corpo inzialmente in posizione verticale, avremo che questa grandezza corrisponde alla congiungente dal centro del sistema di riferimento al punto P_c, che equivale a dire che

$$l_c = 0.55 \bullet h_c + \frac{h_b}{2}$$

- l_b: spostamento verso il basso, lungo l'asse z, del baricentro dello chassis. Da specifiche del progetto sappiamo che questa grandezza ha valore (con segno negativo) di:

$$l_b = 0.1m$$

- β: angolo formato con la verticale dalla congiungente tra il centro del sistema di riferimento e il punto P_a. Si ricava, con un semplice approccio trigonometrico, che l'angolo in questione ha questa forma

$$\arctan\left(\frac{\frac{w_b}{2}}{\frac{h_a}{2} + \frac{h_b}{2}}\right)$$

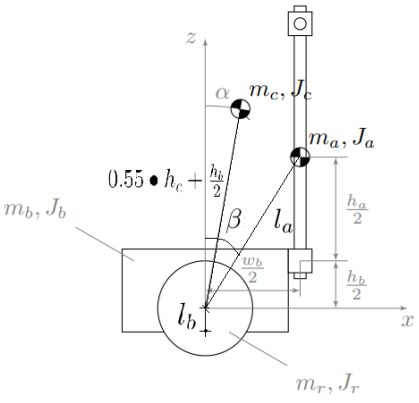

Figure 2.2: Lunghezze di supporto

2.3 Calcolo componenti dinamiche e potenziali per ogni corpo rigido del sistema

Per ognuno dei corpi rigidi definiti in precedenza siamo andati appunto a calcolare:

- **Coordinate spaziali P** espresse nel sistema di riferimento XZ. A queste due coordinate cartesiane ne va aggiunta una terza, relativa alle coordinate angolari (per poter tener così conto dei contirbuti inerziali);

- **Vettore delle velocità V** $\rightarrow$ vettore 3×1
- **Matrice delle masse M** $\rightarrow$ matrice 3×3
- **Energia cinetica T** $\rightarrow \frac{1}{2} \bullet V^T \bullet M \bullet V$
- **Energia potenziale U**
- **Lagrangiana *parziale* L**

2.3.1 Asta

- $P_a = \begin{pmatrix} r\,\phi(t) + l_a \sin(\beta + \theta(t)) \\ l_a \cos(\beta + \theta(t)) \\ \theta(t) \end{pmatrix}$
- $V_a = \begin{pmatrix} r\,\dot{\phi}(t) + l_a \cos(\beta + \theta(t))\ \dot{\theta}(t) \\ -l_a \sin(\beta + \theta(t))\ \dot{\theta}(t) \\ \dot{\theta}(t) \end{pmatrix}$
- $M_a = \begin{pmatrix} m_a & 0 & 0 \\ 0 & m_a & 0 \\ 0 & 0 & m_a\,{l_a}^2 + J_a \end{pmatrix}$
- $T_a = m_a\,{l_a}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_a\,r^2\left(\dot{\phi}(t)\right)^2}{2} + \frac{J_a\left(\dot{\theta}(t)\right)^2}{2} + m_a \cos(\beta + \theta(t))\ l_a\,r\,\dot{\theta}(t)\,\dot{\phi}(t)$
- $U_a = g\,l_a\,m_a \cos(\beta + \theta(t))$
- $L_a = m_a\,{l_a}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_a\,r^2\left(\dot{\phi}(t)\right)^2}{2} + \frac{J_a\left(\dot{\theta}(t)\right)^2}{2} + m_a \cos(\beta + \theta(t))\ l_a\,r\,\dot{\theta}(t)\,\dot{\phi}(t) - g\,m_a \cos(\beta + \theta(t))\ l_a$

2.3.2 Chassis (base)

- $P_b = \begin{pmatrix} r\,\phi(t) - l_b \sin(\theta(t)) \\ -l_b \cos(\theta(t)) \\ \theta(t) \end{pmatrix}$
- $V_b = \begin{pmatrix} r\,\dot{\phi}(t) - l_b \cos(\theta(t))\ \dot{\theta}(t) \\ l_b \sin(\theta(t))\ \dot{\theta}(t) \\ \dot{\theta}(t) \end{pmatrix}$
- $M_b = \begin{pmatrix} m_b & 0 & 0 \\ 0 & m_b & 0 \\ 0 & 0 & m_b\,{l_b}^2 + J_b \end{pmatrix}$
- $T_b = m_b\,{l_b}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_b\,r^2\left(\dot{\phi}(t)\right)^2}{2} + \frac{J_b\left(\dot{\theta}(t)\right)^2}{2} - m_b \cos(\theta(t))\ l_b\,r\,\dot{\theta}(t)\,\dot{\phi}(t)$
- $U_b = -g\,l_b\,m_b \cos(\theta(t))$
- $L_b = m_b\,{l_b}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_b\,r^2\left(\dot{\phi}(t)\right)^2}{2} + \frac{J_b\left(\dot{\theta}(t)\right)^2}{2} - m_b \cos(\theta(t))\ l_b\,r\,\dot{\theta}(t)\,\dot{\phi}(t) + g\,m_b \cos(\theta(t))\ l_b$

2.3.3 Utente

- $P_c = \begin{pmatrix} r\,\phi(t) + l_c \sin(\alpha + \theta(t)) \\ l_c \cos(\alpha + \theta(t)) \\ \alpha + \theta(t) \end{pmatrix}$
- $V_c = \begin{pmatrix} r\,\dot{\phi}(t) + l_c \cos(\alpha + \theta(t))\ \dot{\theta}(t) \\ -l_c \sin(\alpha + \theta(t))\ \dot{\theta}(t) \\ \dot{\theta}(t) \end{pmatrix}$
- $M_c = \begin{pmatrix} m_c & 0 & 0 \\ 0 & m_c & 0 \\ 0 & 0 & m_c\,{l_c}^2 + J_c \end{pmatrix}$
- $T_c = m_c\,{l_c}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_c\,r^2 \left(\dot{\phi(t)}\right)^2}{2} + \frac{J_c \left(\dot{\theta(t)}\right)^2}{2} + m_c \cos(\alpha + \theta(t))\ l_c\,r\,\dot{\theta}(t)\,\dot{\phi}(t)$
- $U_c = g\,l_c\,m_c \cos(\alpha + \theta(t))$
- $L_c = m_c\,{l_c}^2 \left(\dot{\theta}(t)\right)^2 + \frac{m_c\,r^2 \left(\dot{\phi(t)}\right)^2}{2} + \frac{J_c \left(\dot{\theta(t)}\right)^2}{2} + m_c \cos(\alpha + \theta(t))\ l_c\,r\,\dot{\theta}(t)\,\dot{\phi}(t) - g\,m_c \cos(\alpha + \theta(t))\ l_c$

2.3.4 Ruota

- $P_r = \begin{pmatrix} r\,\phi(t) \\ 0 \\ \phi(t) \end{pmatrix}$
- $V_r = \begin{pmatrix} r\,\dot{\phi}(t) \\ 0 \\ \dot{\phi}(t) \end{pmatrix}$
- $M_r = \begin{pmatrix} m_r & 0 & 0 \\ 0 & m_r & 0 \\ 0 & 0 & J_r \end{pmatrix}$
- $T_r = \frac{\left(m_r\,r^2 + J_r\right) \left(\dot{\phi(t)}\right)^2}{2}$
- $U_r = 0$
- $L_r = \frac{\left(m_r\,r^2 + J_r\right) \left(\dot{\phi(t)}\right)^2}{2}$

2.3.5 Veicolo completo

Una volta trovati le componenti dinamiche dei singoli corpi rigidi, possiamo andare a definire l'energia cinetica e potenziale totale del sistema, per poter poi andare a calcolare l'equazione di Lagrange per l'intero sistema. In sostanza quindi avremo:

$$L = L_a + L_b + L_c + 2 \bullet L_r$$

2.3.6 Note sul calcolo delle componenti dinamiche

- Nel calcolo della matrice di massa abbiamo considerato tre componenti:
  - Componente di massa lungo x;
  - Componente di massa lungo z;

  - Componente di massa rotazionale: dalla meccanica è noto che, un corpo con una certa massa che si trova in uno stato di rotazione, avrà un contirbuto inerziale che dipende dal braccio rispetto al quale avviene la rotazione. Nello specifico, per ogni singolo corpo rigido che è stato preso in esame, abbiamo considerato, secondo il teorema di *Huygens-Steiner*, il quale permette di definire l'inerzia di un corpo come la somma di due diverse componenti:
    * Momento d'inerzia definito rispetto all'asse passante per il centro di massa: questo parametro rappresenta il valore che è fornito dalle specifiche del progetto;
    * prodotto tra la massa m del corpo preso in considerazione e la distanza tra l'asse in esame e quello passante per il centro di massa;

    Questa terza componente è visibile nelle matrici di massa in posizione $(3, 1)$: si sottolinea come invece, per la matrice 2.3.4, non sia presente la componente esplicitata dal teorema di *Huygens-Steiner* per il fatto che il centro di massa della ruota coincide con quello del sistema di riferimento XZ;

- Nel calcolo simbolico in ambiente *Matlab* siamo andati ad utilizzare le funzionalità di *collect* e *simplify*, per permettere di ridurre e semplificare le equazioni. Nello specifico, con il comando simplify, tramite l'opzione *Steps*, siamo andati a settare il numeor di step che l'algoritmo di calcolo simbolico andrà a seguire per poter ridurre e semplificare il maggior numero di termini ($\uparrow$ *Steps*, $\uparrow$*Compattezza eq symb*);

- Le ruote, nel calcolo della dinamica completa del VAB, sono state considerate con un contirbuto doppio;

- Nello studio dello chassis (base), abbiamo seguito questo approccio. Il baricentro della base stessa sappiamo essere posizionato ad una quota differente rispetto al centro del sistem di coordinate XZ preso come riferimento.

  Per questo motivo il suo contributo in termini cinetici e potenziali dipende dal valore dell'inclinazione dello chassis stesso, ovvero dal valore dell'angolo *caratterizzante* il sistema θ: questo concetto è evidenziato in figura 2.3.

  L'aggiunta di π al valore di θ è necessaria per poter rendere sensibile i valori di energia cinetica e potenziale al *quadrante* in cui si trova ad essere posizionato il centro di amssa della base stessa (P_b).

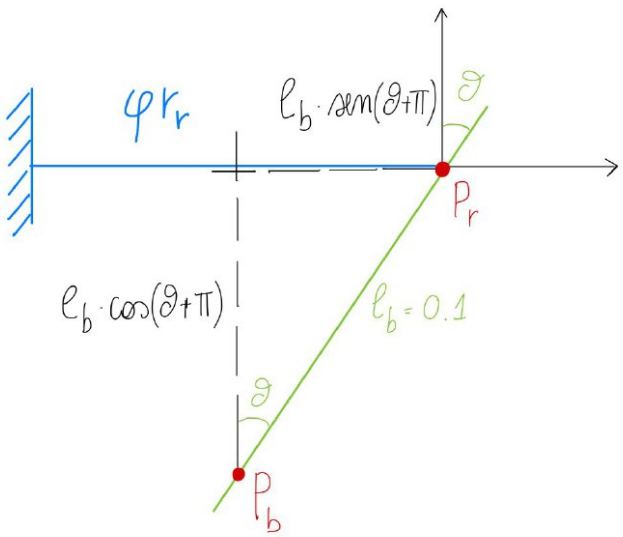

Figure 2.3: Ragionamento per il calcolo dei contributi dinamici dello chassis

3

Equazioni del moto

Una volta definita la dinamica del sistema ed ottenuta quindi l'equazione di Lagrange che ne caratterizza il comportamento, andiamo a ricavare le equazioni del moto, le quali consentono di definire l'andamento delle *coordinate libere* (scelte in fase inziale di progetto) che sappiamo essere

- $\phi = q_1 =$ *angolo di rotazione delle ruote*
- $\theta = q_2 =$ *angolo di inclinazione dello chassis*

In particolare possiamo definire le cosidette *equazioni di Eulero-Lagrange*, ovvero un insieme di n equazioni differenziali (con n pari al numero di coordinate libere del sistema), la cui risoluzione fornisce le equazioni del moto del sistema.

$$\frac{d}{\mathrm{dt}}\left(\frac{\partial}{\partial \dot{q}}L\right) - \frac{\partial}{\partial q}L = \mathrm{torque}$$

Sono da effettuare alcune annotazioni sugli addenti presenti nell'equazione 3:

- q rappresenta la coordinata libera (nel nostro caso sarà θ e ϕ);
- al secondo membro della lagrangiana TODO
- molto importante ai fini della correttezza delle equazioni del moto è il valore da assegnare a *torque*; infatti esso varia a seconda che il sistema agisca sulla coordinata libera q a valle o a monte della trasmissione. Nello specifico, si osserva che il motore è composto da due parti: lo statore e il rotore. Esse trasmettono una coppia uguale e inversa a ciò cui sono collegati: lo statore, posizionato nella base del Segway, influenza la coordinata θ ed esercita una coppia sullo chassy pari ad una generica $-2C_m$; il fattore moltiplicativo 2 è dovuto al fatto che nella base sono presenti due motori. Parimenti, il rotore trasmette all'albero motore una coppia C_m. A valle della trasmissione si misura dunque una coppia $\frac{C_m}{\tau}$ dovuta alla presenza della trasmissione; questa coppia ridotta all'albero dell'utilizzatore è la coppia che in definitiva va ad agire sulle ruote e quindi sulla coordinata ϕ

Si ottiene dunque il seguente sistema di equazioni:

$$\frac{d}{\mathrm{dt}}\left(\frac{\partial}{\partial \dot{\theta}}L\right) - \frac{\partial}{\partial \theta}L = -2\mathrm{C_m}$$
$$\frac{d}{\mathrm{dt}}\left(\frac{\partial}{\partial \dot{\phi}}L\right) - \frac{\partial}{\partial \phi}L = \frac{\mathrm{C_m}}{\tau}$$

dove τ è il rapporto di trasmissione calcolato come segue:

$$\tau_1 = 0.1$$
$$\tau_2 = \frac{Z_in}{Z_out} = \frac{22}{26}$$
$$\tau = \tau_1 \cdot \tau_2 = 0.085$$

La risoluzione delle equazioni 3, permette di ottenere due equazioni differenziali del secondo ordine che il comando *matlabFunction()* va a scrivere in una funzione di matlab: questa funzione avrà come input i valori da cui dipendono le equazioni differenziali e moltiplicandoli tra di loro da come output il valore di $\ddot{\theta}$ e $\ddot{\phi}$. TODO (mettere le dipendende di theta pp e phi pp)

3.0.1 Linearizzazione

TODO linearizzazione attorno alla massa 70 kg Il passo successivo che è stato svolto riguarda la linearizzazione; linearizzare è importante poiché il controllore viene progettato sul sistema lineare e poi testato sul sistema reale e quindi non lineare.

Per tale ragione è necessario innanzitutto calcolare l'equilibrio del sistema, cioè il punto in cui $\ddot{\theta} = 0$:

$$\begin{pmatrix} -0.01174364 - 7.105427\text{e-}15\,\mathrm{i} \\ 3.129849 - 7.105427\text{e-}15\,\mathrm{i} \end{pmatrix}$$

I risultati ottenuti sono, circa, numeri naturali e rispondono a quanto ci aspettavamo:

- il sistema ha due equilibri, il primo $\theta = -0.01174364$ ci si aspetta che sia instabile in quanto il baricentro del sistema V.A.B. più Uomo è sopra all'asse delle ruote
- $\theta = 3.129849$ è un equilibrio stabile poiché il baricentro sta sotto l'asse delle ruote e un eventuale disturbo, dopo un transitorio, risulterebbe avere effetto nullo sul sistema.

Ovviamente, ai fini del controllo, si è linearizzato attorno al primo equilibrio; non avrebbe senso cpntrollare il sistema quando questo risulta capovolto. Il vettore degli stati e delle uscite sono i seguenti:

$$x = \begin{bmatrix} x_1 \to \phi \\ x_2 \to \dot{\phi} \\ x_3 \to \theta \\ x_4 \to \dot{\theta} \end{bmatrix}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

e vengono definite anche alcune variabili di supporto:

$$\dot{x_1}(t) = \bar{x_2} = 0 \to f_1$$
$$\dot{x_2}(t) = \ddot{\phi} \to f_2$$
$$\dot{x_3}(t) = \bar{x_4} = 0 \to f_3$$
$$\dot{x_4}(t) = \ddot{\theta} \to f_4$$
$$y_1\ (t) = \phi = x_1 \to g_1$$
$$y_2\ (t) = \theta = x_2 \to g_2$$

ricordando che un generico sistema dinamico può essere scritto come:

$$G(s) = C \bullet (\mathrm{sI} - A)^{-1} \bullet B + D$$

allora:

$$A = \begin{bmatrix} \frac{\partial}{\partial x_1} f_1 & \frac{\partial}{\partial x_2} f_1 & \frac{\partial}{\partial x_3} f_1 & \frac{\partial}{\partial x_4} f_1 \\ \frac{\partial}{\partial x_1} f_2 & \frac{\partial}{\partial x_2} f_2 & \frac{\partial}{\partial x_3} f_2 & \frac{\partial}{\partial x_4} f_2 \\ \frac{\partial}{\partial x_1} f_3 & \frac{\partial}{\partial x_2} f_3 & \frac{\partial}{\partial x_3} f_3 & \frac{\partial}{\partial x_4} f_3 \\ \frac{\partial}{\partial x_1} f_4 & \frac{\partial}{\partial x_2} f_4 & \frac{\partial}{\partial x_3} f_4 & \frac{\partial}{\partial x_4} f_4 \end{bmatrix}$$

$$B = \begin{bmatrix} \frac{\partial}{\partial u} f_1 \\ \frac{\partial}{\partial u} f_2 \\ \frac{\partial}{\partial u} f_3 \\ \frac{\partial}{\partial u} f_4 \end{bmatrix}$$

$$C = \begin{bmatrix} \frac{\partial}{\partial x_1} g_1 & \frac{\partial}{\partial x_2} g_1 & \frac{\partial}{\partial x_3} g_1 & \frac{\partial}{\partial x_4} g_1 \\ \frac{\partial}{\partial x_1} g_1 & \frac{\partial}{\partial x_2} g_1 & \frac{\partial}{\partial x_3} g_1 & \frac{\partial}{\partial x_4} g_2 \end{bmatrix}$$

$$D = \begin{bmatrix} \frac{\partial}{\partial u} g_1 & \frac{\partial}{\partial u} g_2 \end{bmatrix}$$

Sostituendo i valori numerici dei vari simboli si ottengono le seguenti matrici:

$$\begin{pmatrix} \frac{2.75}{s^2} - \frac{1.749\text{e+}13}{2.392\text{e+}13\, s^2 - 4.398\text{e+}12\, s^4} \\ \frac{2.75}{s} - \frac{1.749\text{e+}13}{2.392\text{e+}13\, s - 4.398\text{e+}12\, s^3} \\ -\frac{1.149\text{e+}12}{4.398\text{e+}12\, s^2 - 2.392\text{e+}13} \\ -\frac{1.149\text{e+}12\, s}{4.398\text{e+}12\, s^2 - 2.392\text{e+}13} \end{pmatrix}$$

Si può notare come la prima e la seconda riga della matrice differiscano solo per un fattore derivativo, così come la terza e quarta riga; questo è ovvio ed atteso in quanto x_2 è a derivata di x_1 e x_4 la derivata di x_3

Part II

Simulink

4

Simulazione

Data la straordinarietà degli eventi che erano in corso dal punto di vista sanitario nel nostro paese si è reso necessario svolgere gran parte del lavoro in modalità a distanza e quindi senza la possibilità di testare il modello matematico appena ottenuto e i successivi risultati dovuti all'azione di controllo sul V.A.B. Il lavoro quindi è stato svolto per la maggior parte sfruttando il tool *Simulink* di Matlab che è, in poche parole, un risolutore di equazioni differenziali.

Abbiamo così adottato una metodologia di lavoro basata su prototipi sempre più simili a quello che dovrebbe essere il sistema reale.

4.1 Simulazione del sistema reale

Il primo compito che abbiamo risolto è stato quello di implementare le equazioni differenziali ottenute nel capitolo precedente:

- $\ddot{\phi} = f_{\ddot{\phi}}(M_c, \theta, \dot{\theta}, C_m)$
- $\ddot{\theta} = f_{\ddot{\theta}}(M_c, \theta, \dot{\theta}, C_m)$

Dove M_c sarebbe la massa del passeggero, $\theta e \dot{\theta}$ lo stato del sistema e C_m la coppia erogata dal motore. Si può notare come entrambe le equazioni differenziali siano indipendenti dalla coordinata libera ϕ.

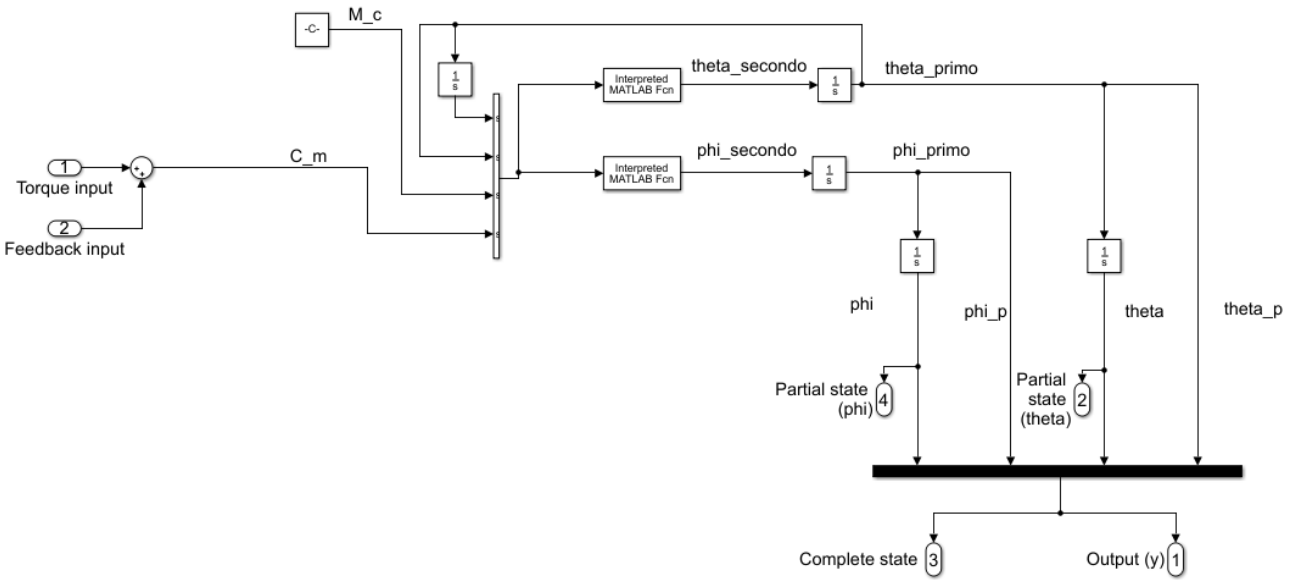

Figure 4.1: Implementazione simulink delle equazioni differenziali

In Fig.4.1 le *interpreted function* altro non sono che $f_{\ddot{\phi}}$ e $f_{\ddot{\theta}}$. A valle di esse sono presenti degli integratori che permettono di ottenere lo stato x completo del sistema. Si può facilmente notare

come $\dot{\theta}e\theta$ siano collegate direttamente all'input delle *interpreted function*. Si è dunque proceduto a simulare il sistema per verificare la bontà di quanto ottenuto; in particolare, il sistema in anello aperto, dovrebbe oscillare all'infinito vista la mancanza di attriti nel modello.
TODO grafico e foto sistema simulink non lineare anello aperto e chiuso
Si è inoltre creato un altro modello sfruttando il sistema lineare ottenuto prima con l'obbiettivo di semplificare il problema e di velocizzare le simulazioni con lo scopo di testare rapidamente nuove tecniche di controllo che se avessero dato esito positivo sul modello lineare sarebbero poi state testate sul simulink che imita il comportamento reale del sistema.

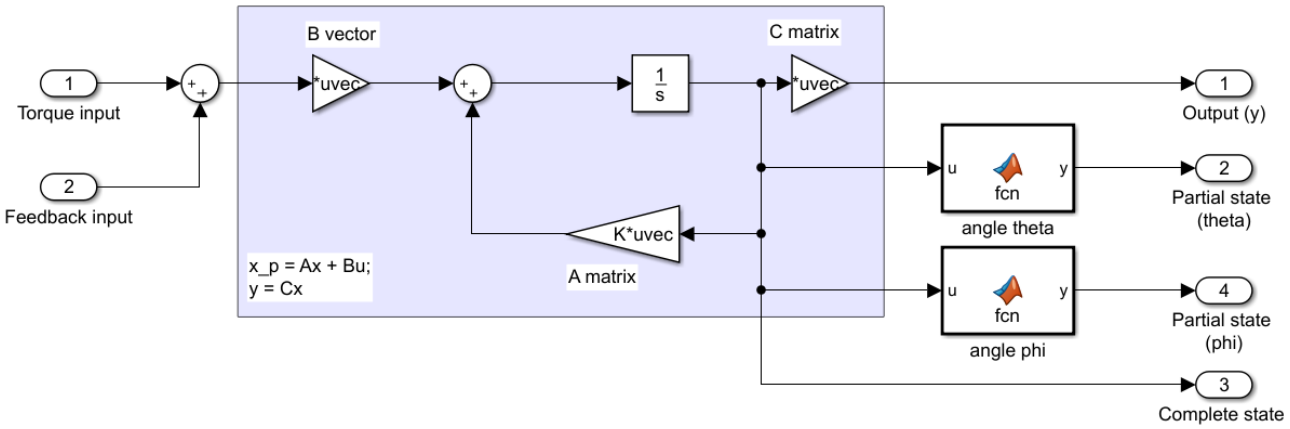

Figure 4.2: Implementazione simulink del sistema linearizzato

TODO grafico e foto sistema lineare anello aperto e chiuso
In questo caso, la simulazione in anello aperto mostra che il sistema diverge; questo perché la linearizzazione ha senso attorno al punto di equilibrio da cui è stata ottenuta, distante da quel punto il sistema lineare non approssima più il sistema reale ed anche un eventuale controllo ottenuto da esso non garantisce buone performance distante da quel punto.

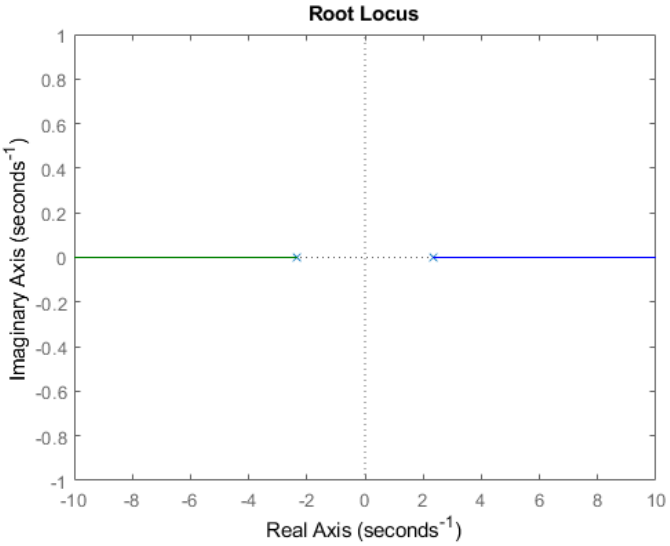

Figure 4.3: Posizione poli in anello aperto

Come si vede in Fig.4.3, ci sono due poli reali, uno negativo e uno positivo; questo dimostra come il sistema sia instabile in anello aperto e necessiti di controllo.

4.2 Controllore

Il problema della scelta del controllore è stato risolto mediante l'utilizzo dell'assegnazione degli autovalori ottenuti tramite retroazione dello stato; si affronta quindi un problema di regolazione: il moto del sistema è composto completamente dal moto libero che si vuole controllare e annullare in un tempo a piacere. Il posizionamento dei poli, e quindi la scelta del guadagno del regolatore, va eseguita sul solo sistema lineare:

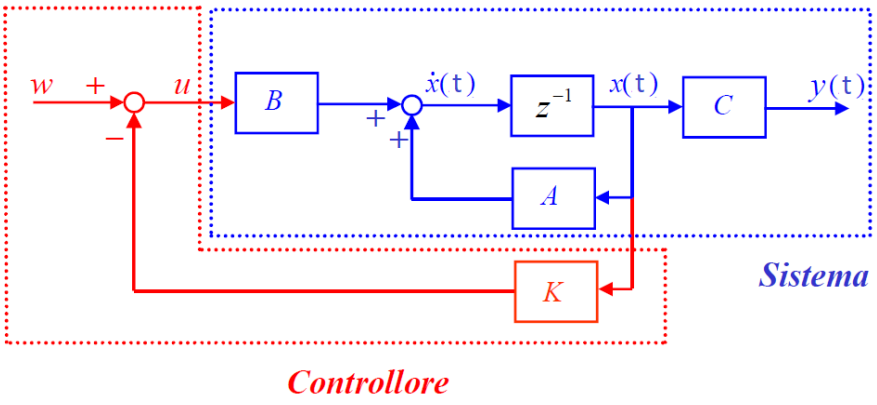

Figure 4.4: Schema concettuale del controllo,[1]

Ciò che si nota in Fig.4.4 può essere riscritto matematicamente:

$$\begin{cases} x\,(t) = Ax(t) + Bu\,(t) \\ y(t) = Cx(t) + Du\,(t) \end{cases}$$

$$u(t) = -Kx(t) + w(t)$$

Dunque:

$$x(t) = Ax(t) + Bu(t) = Ax(t) + B\big(-Kx(t) + w(t)\big)$$

$$= Ax(t) - BKx(t) + Bw\,(t) = (A - BK)x(t) + Bw(t)$$

$$A_{closedloop} = A - BK$$

Per scegliere il valore di guadagno del controllore, è necessario scegliere la posizione desiderata dei poli:

$$G(S) = \frac{\omega_n^{\,2}}{s^2 + 2\xi\omega_{\,n} s + \omega_n^{\,2}}$$

in cui:

$$\xi = 0.7$$

$$\omega_{\,n} = 2 \bullet \pi \bullet f_{\text{propria}}$$

Dovendo posizionare due coppie di poli complessi coniugati si sono scelte due frequenze ad una decade di distanza

$$f_{\text{propria}\theta} = 0.2$$

$$f_{\text{propria}\phi} = 0.02$$

Per motivi esterni il motore ha una coppia massima molto limitata ed è stato dunque necessario posizionare i poli in modo che non fossero troppo veloci.

$$polo_\theta = \begin{pmatrix} -\frac{7\pi}{250} + \frac{\pi\sqrt{51}\,\mathrm{i}}{250} \\ -\frac{7\pi}{250} - \frac{\pi\sqrt{51}\,\mathrm{i}}{250} \end{pmatrix}$$

$$polo_\phi = \begin{pmatrix} -\frac{7\pi}{25} + \frac{\pi\sqrt{51}\,\mathrm{i}}{25} \\ -\frac{7\pi}{25} - \frac{\pi\sqrt{51}\,\mathrm{i}}{25} \end{pmatrix}$$

Questi quattro poli si può notare che abbiano parte reale negativa; la stabilità in questo modo è raggiunta.
Con il comando *place* di Matlab si ottine dunque:

$$K = [-0.0023, -0.0278, -28.1397, -7.7022]$$

Il sistema in anello chiuso ha i seguenti poli:

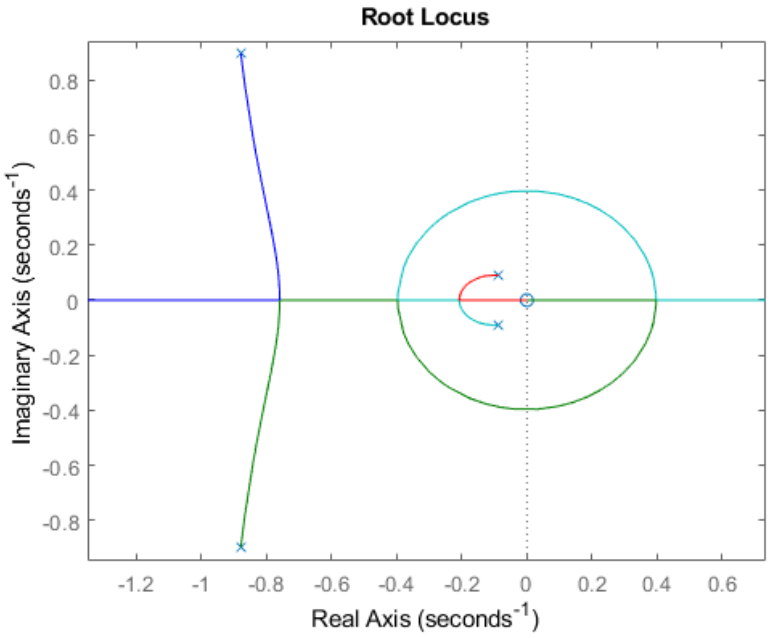

Figure 4.5: Root locus del sistema in anello chiuso

Part III

Controllo

5

OPC - UA

Per quanto riguarda la parte di controllo, il sistema presenta un articolato insieme di controllori inter-operanti tra di loro: nello specifico ad ogni singolo controllore sono affidate delle mansioni ben specifiche, tutte ovviamente volte al controllo e alla stabilizzazione del *veicolo auto bilanciato*.

In questa fase dello sviluppo del progetto, siamo andati ad implementare parte del codice che verrà installato, in un secondo momento, a bordo del raspberry: esso infatti svolge, all'interno del sistema (come si vede in figura 5.1) una comunicazione a due direzioni, che ne determinano due comportamenti differenti:

- Come **server** per la parte di comunicazione *OPC-UA* (per il settaggio dei guadagni);
- Come **master** nella comunicazione seriale verso Arduino (per quanto riguarda invece la gestione dell'algoritmo di controllo);

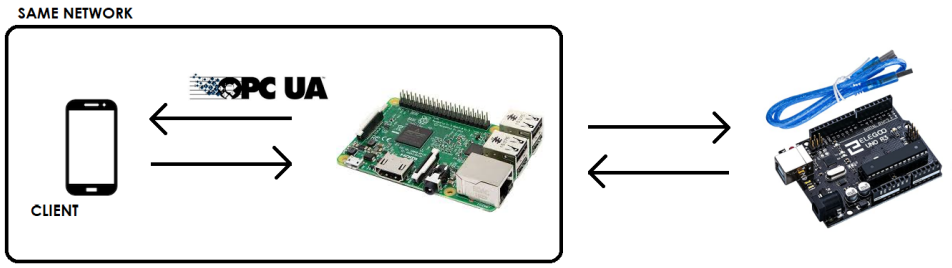

Figure 5.1: Schema di massima dell'utilizzo di Raspberry Pi 3

In questa fase abbiamo quindi sviluppato la parte relativa all'utilizzo di *Raspberry Pi 3* come *server OPCUA*.

5.1 Idea base OPC-UA

L'*Open Platform Communications Unified Architecture* (OPC UA) è un protocollo di comunicazione automatico per l'automazione industriale.

OPC UA sostituisce il protocollo *OPC Classic*, conservando tutte le funzionalità del predecessore. Poiché OPC Classic era stato costruito su una tecnologia Microsoft detta modello a oggetti per componenti distribuiti, era vincolato a Microsoft, ma questa caratteristica è diventata sempre più limitante.

OPC UA risulta completamente interoperabile tra i diversi sistemi operativi usati, aggiungendosi a Windows e alle tecnologie industriali come i PLC, inoltre comprende Linux, iOS e anche sistemi operativi per dispositivi mobili come Android. Consentire al maggior numero di dispositivi possibile di comunicare contribuisce al progresso dell'IoT.

Queste caratteristiche di **interoperabilità** sono state sfruttate al massimo in questo contesto, potendo così creare, in maniera semplice e veloce, una comunicazione tra differenti tipologie e famiglie di dispositivi.

5.2 OPC-UA e V.A.B.

Nel contesto del progetto del *veicolo auto bilanciato*, siamo andati ad utilizzare il protocollo di comunicazione *OPC-UA* come supporto per il tuning dei parametri relativi al **gain** del controllore, ovvero ai parametri del vettore K, che abbiamo chiamato (all'interno dello script di Python):

- **K_phi**
- **K_phi_p**
- **K_theta**
- **K_theta_p**

Nello specifico, lo scambio di parametri tra server e controllore avviene tramite unfile *.txt*, che permette, in maniera semplice e immediata, di implementare uno scambio di informazioni tra il server *OPC-UA* strutturato in Python e l'ambiente *real-time* introdotto a bordo del controllore *Raspbery Pi 3*.

In particolare abbiamo due files:

- **Un file temporaneo** *("GainParametersToController.txt")* utilizzato come pipeline per il passaggio dei parametri tra server e controllore. Questo risulta essere un file temporaneo che viene ad essere cancellato e ricreato ogni qualvolta che il server viene spento e successivamente riaccesso. Nello specifico, ad ogni riaccensione, i valori iniziali di questo file, vengono settati con gli stessi valori presenti nel file *definitivo* (qualora quest'ultimo esista già: in caso contrario si procede con un inizializzazione dei parametri a 0);
- **Un file definitivo** *("GainParametersConfirmed.txt")* il quale invece viene creato una e una sola volta e sul quale poi vengono salvati i parametri che saranno poi letti all'accensione successiva del server ed utilizzati come parametri iniziali per il controllore.

Questi file possono essere settati con i parametri presenti nel server che sono visibili in figura 5.2, nello specifico:

- **Submit change to controller** permette di scrivere i valori dei gains sul file *"GainParametersToController.txt"*;
- **Store definitively in file** permette di scrivere i valori dei gains sul file *"GainParametersConfirmed.txt"*;
- **SHUT DOWN SERVER** permette invece di spegnere il server e di cancellare successivamente il file temporaneo.

Abbiamo racchiuso in figura 5.3 il funzionamento di massima del codice lato server: codice che siamo andati a testare utilizzando un'apposita app per smartphone Android (OPC-UA Android client).

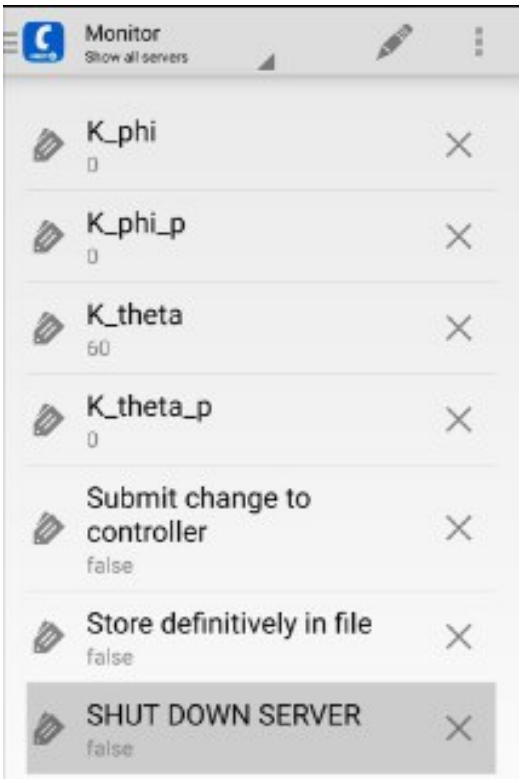

Figure 5.2: Parametri impostabili lato client

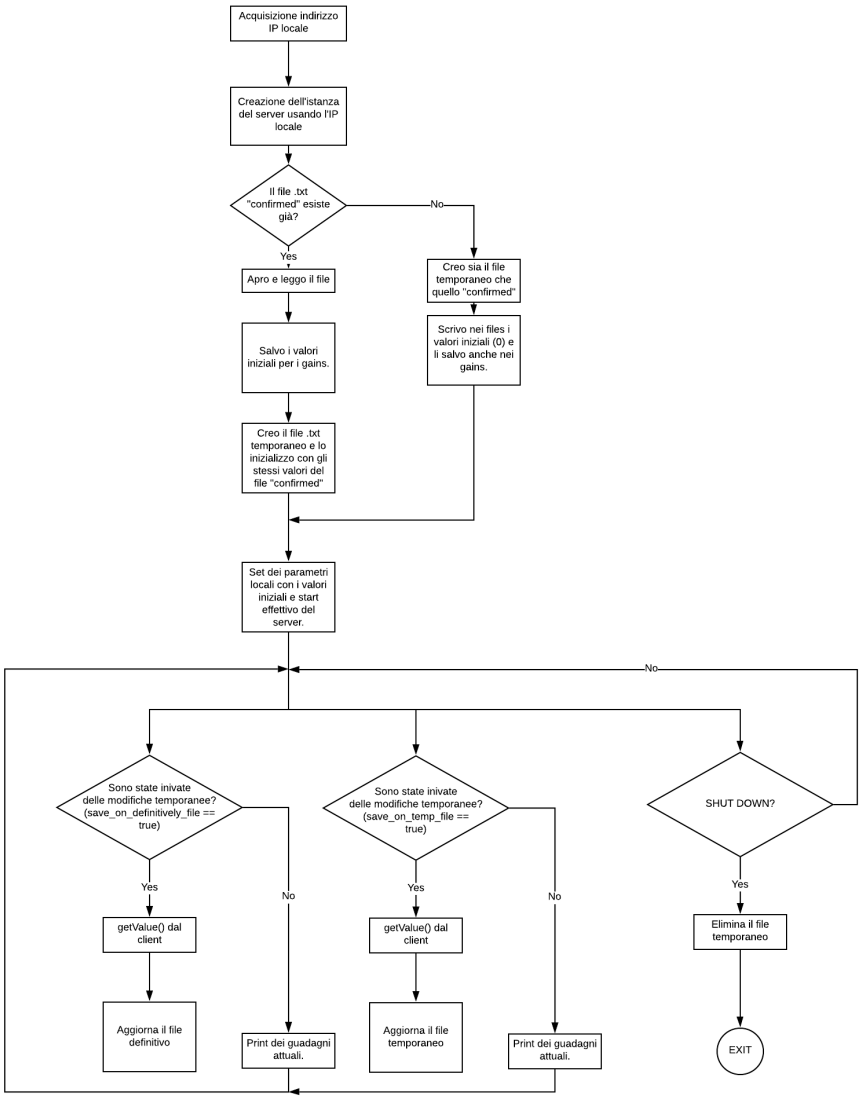

Figure 5.3: Diagramma rappresentante le funzionalità del server

6

Rumore

Piattaforma inerziale –> PSD Questione anti windup –> coppia già limitata quindi nessun vincolo di saturazione

Bibliography

[1] *Controllo tramite retroazione dello stato,Controlli automatici, Fabio Previdi* https://cal.unibg.it/wp-content/uploads/controlli_automatici/Lez06.pdf