POLITECNICO DI MILANO
Corso di Laurea in Ingegneria Informatica
Dipartimento di Elettronica e Informazione

Progettazione e realizzazione di una base robotica bilanciante su ruote

AI & R Lab
Laboratorio di Intelligenza Artificiale
e Robotica del Politecnico di Milano

Relatore: Prof. Andrea Bonarini

Tesi di Laurea di:
Martino Migliavacca, matricola 682460

Anno Accademico 2008-2009

Sommario

Nell'ambito della robotica mobile destinata all'interazione con l'uomo e al movimento in ambienti domesitici sono state apprezzate le soluzioni basate sulla costruzioni di robot bilancianti su ruote, in grado di mantenere l'equilibrio.
L'obiettivo della tesi è progettare e realizzare una base robotica bilanciante di dimensioni tali da poter interagire con l'uomo e trasportare un determinato carico utile, capace di muoversi in ambiente domestico inseguendo una determinata traiettoria.
Il lavoro presentato riguarda l'intero sviluppo del sistema, a partire dalla costruzione di un modello del sistema e dalla progettazione del telaio, delle componenti meccaniche e dalla realizzazione dell'elettronica necessaria per il controllo del robot e per l'acquisizione dei dati sensoriali. Sono stati in seguito implementati degli algoritmi in grado di stimare l'inclinazione rispetto alla verticale, di mantenere in equilibrio il robot e di inseguire un profilo di velocità.
Le prove sperimentali effettuate mostrano ottimi risultati nella stima dell'inclinazione del robot e capacità di bilanciamento utilizzando i diversi algoritmi di controllo, dei quali sono evidenziate le differenti caratteristiche e identificati gli elementi su cui intervenire per migliorarne il comportamento.

Ringraziamenti

Ringrazio innanzitutto la mia famiglia, che da un lato mi ha incentivato a coltivare i miei interessi, e dall'altro ha sopportato le conseguenze a cui, i miei interessi, hanno portato. In particolare ringrazio mio padre per avermi regalato, all'età di nove anni, un gioco che non dimenticherò mai, e mia madre per avermi consentito, dall'età di nove anni, di trasformare la mia stanza in una sorta di laboratorio.
Ringrazio Serena, lei sa perchè. Tutti sanno perchè. Sono una persona molto fortunata.
Ringrazio i miei amici, che sono restati tali nonostante i miei discorsi, col passare degli anni, siano diventati sempre più monotematici.
Ringrazio tutto il gruppo AIRLab, il cui aiuto è stato preziosissimo in tante occasioni. La passione che caratterizza le persone presenti nel laboratorio, e il clima che in esso si respira, hanno reso incredibilmente piacevoli i mesi di lavoro durante cui la tesi è stata sviluppata.
Un ringraziamento va al Prof. Andrea Bonarini, senza il quale un posto come l'AIRLab non sarebbe mai esistito, o perlomeno sarebbe stato molto differente da come l'ho conosciuto, e all'Ing. Marcello Restelli, grazie a cui la mia curiosità ha trovato molti nuovi argomenti da esplorare.

Indice

Capitolo 1

Introduzione

"Input. Necessito input."

Numero 5 - Corto Circuito

La ricerca di soluzioni per la costruzione robot adatti a muoversi in ambienti domestici e a cooperare con l'uomo è un argomento di studio molto diffuso negli ultimi anni, e molte aziende stanno investendo in questo settore con la convinzione che nel prossimo futuro la robotica rappresenterà uno dei mercati più interessanti.
Tra i vari approcci proposti sono stati apprezzati, per semplicità costruttiva, efficienza energetica e ridotti costi di realizzazione quelli basati sulla costruzione di robot bilancianti su ruote, in grado di mantenersi in equilibrio e, grazie alla ridotta impronta a terra, di muoversi con agilità in spazi ristretti.
L'obiettivo della tesi è progettare e realizzare una base robotica bilanciante di dimensioni tali da poter interagire con l'uomo e trasportare dispositivi accessori, capace di muoversi in ambiente domestico inseguendo una determinata traiettoria.
Il lavoro presentato riguarda l'intero sviluppo del sistema, a partire dalla costruzione di un modello del sistema e dalla progettazione del telaio, delle componenti meccaniche e dalla realizzazione dell'elettronica necessaria per il controllo del robot e per l'acquisizione dei dati sensoriali. Sono stati in seguito implementati degli algoritmi in grado di stimare l'inclinazione rispetto alla verticale, di mantenere in equilibrio il robot e di inseguire un profilo di velocità.

La fase preliminare del progetto consiste nello studio analitico del problema, che ha portato alla realizzazione di un modello del sistema al fine di analizzare le relazioni tra forze applicate e moto del robot. Il modello ricavato è stato inserito in un ambiente di simulazione che ha permesso di dimensionare le componenti necessarie e, in seguito, di studiare il comportamento dei diversi algoritmi di controllo.

Il lavoro svolto procede con la progettazione delle parti meccaniche, ricorrendo a soluzioni che soddisfino le richieste di modularità del sistema e di utilizzo di componenti il più possibile standard. Dopo aver realizzato il telaio sono state progettate le ruote e la trasmissione, ed infine sono stati scelti gli attuatori.

Il progetto elettronico ha riguardato la scelta dei sensori utilizzati per la stima dello stato del sistema, il condizionamento dei segnali da essi estratti e la progettazione della logica di controllo basata su microcontrollore. Sono state infine sviluppate di due diverse soluzioni di potenza per pilotare i motori elettrici.

Completata la realizzazione dell'hardware ci si è concentrati sul problema del controllo. Sono state dapprima implementate soluzioni basate sui controllori classici di tipo PID e LQR, in grado di mantenere in equilibrio il robot e di inseguire un profilo di velocità. In seguito è stato studiato un approccio basato sull'apprendimento per rinforzo, tramite cui il sistema apprende una politica di controllo osservando l'effetto delle azioni che può intraprendere, senza che sia necessario fornire un modello del moto.

L'ultima parte del lavoro consiste nello svolgimento di esperimenti relativi alla stima dell'inclinazione del robot tramite i dati forniti dai sensori e nell'analisi delle prestazioni dei differenti algoritmi di controllo. In particolare le informazioni ricavate dal filtraggio dei dati sensoriali sono state confrontate con le informazioni reali, analizzandone sia l'accuratezza che il ritardo. Le diverse tecniche di controllo sono state testate sul robot registrandone il comportamento, permettendo di osservare prestazioni e differenze delle soluzioni implementate.

I risultati ottenuti mostrano come attraverso un opportuno filtraggio sia possibile estrarre il valore dell'angolo del robot rispetto alla verticale partendo da sensori rumorosi e dati che contengono informazioni non legate all'inclinazione. I controllori classici implementati mostrano qualità differenti, in quanto l'algoritmo PID, che consente di eseguire una taratura pratica, per-

mette di mantenere con facilità l'equilibrio ma difficilmente si può applicare al problema dell'inseguimento di un profilo di velocità; il controllore LQR permette invece di inseguire un setpoint su tutte le variabili di stato ma richiede un controllo in coppia dei motori molto preciso. L'approccio basato sull'apprendimento per rinforzo ha prodotto risultati positivi, permettendo al robot di imparare a mantenere l'equilibrio, ma i ritardi intrinseci del sistema, legati alla lentezza della risposta meccanica alle sollecitazioni, richiedono la ricerca di soluzioni alternative per essere gestiti.
Gli sviluppi futuri riguardano quindi l'affinamento dell'elettronica di controllo dei motori, per permettere ai controllori classici di raggiungere prestazioni migliori, e la ricerca di tecniche di apprendimento per rinforzo in grado di gestire i ritardi ineliminabili da cui il sistema e' affetto.

1.1 Struttura della tesi

Nella sezione 2 è presentato lo stato dell'arte con riferimenti alle soluzioni realizzate in passato e ad articoli inerenti agli argomenti trattati.
Nella sezione 3 si illustra l'impostazione del problema di ricerca, introducendo le nozioni che hanno permesso di ricavare il modello del moto del sistema e la tecnica di filtraggio utilizzata per la stima dell'inclinazione del robot sulla base dei dati forniti dai sensori.
Nella sezione 4 si presenta il progetto meccanico, mostrando le scelte progettuali effettuate.
Nella sezione 5 è riportato il progetto elettronico, relativo alla scelta dei sensori, al condizionamento dei segnali da essi estratti, alla realizzazione della parte logica di controllo e dell'elettronica di potenza necessaria al pilotaggio dei motori.
Nella sezione 6 è presentata l'architettura del software di gestione implementato sul microcontrollore.
Nella sezione 7 si illustrano le differenti tecniche di controllo analizzate, sia basate sui controllori classici sia fondate sul concetto di apprendimento per rinforzo.
Nella sezione 8 sono riportati i risultati sperimentali riguardanti la stima dell'inclinazione, il ritardo dei dati ricavati rispetto ai dati reali e il comportamento delle differenti tecniche di controllo implementate.
Nella sezione 9 sono esposte le conclusioni tratte dall'osservazione delle prove sperimentali e si definiscono gli sviluppi futuri necessari per il miglioramento delle prestazioni ottenute.

Capitolo 2

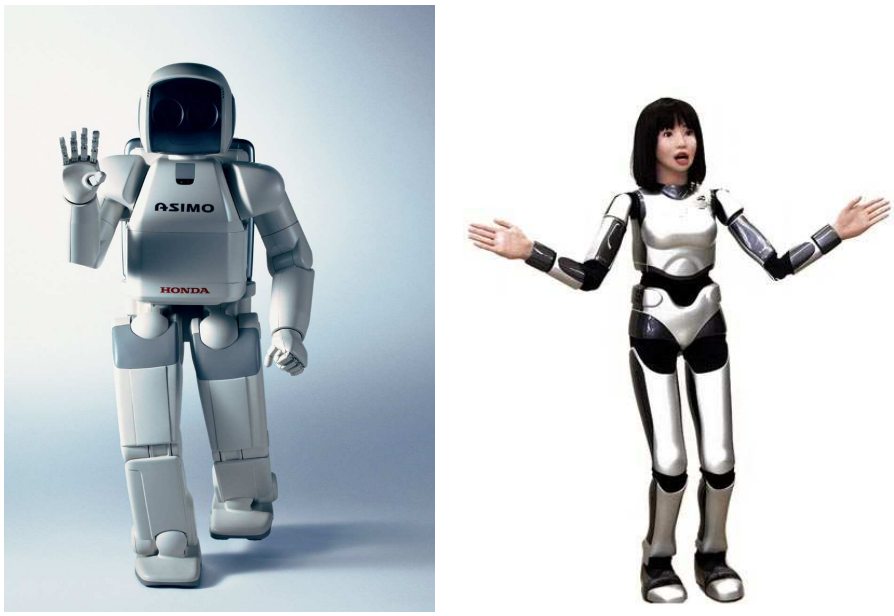
Stato dell'arte

"TUTTI DOVREBBERO POSSEDERE UN ROBOT PERSONALE ROSSUM'S"

R.U.R. Rossum's Universal Robots - Karel Capek, 1920

Il campo dei robot mobili adatti ad operare in ambienti domestici è un argomento di ricerca molto attivo in robotica. L'idea di poter fornire un sussidio all'uomo nello svolgere operazioni comuni, assieme alla convinzione che la robotica rappresenterà il prossimo boom del mercato dell'elettronica, ha portato a studiare e proporre soluzioni differenti per lo sviluppo di robot adatti a cooperare con gli esseri umani. Alcuni gruppi di ricerca hanno intrapreso la strada dei robot umanoidi, che ricalcando il funzionamento del corpo umano sono naturalmente adatti ad operare negli ambienti costruiti per l'uomo e ad interagire con esso. Sono state realizzate diverse piattaforme commerciali, come Asimo di Honda (2000), Qrio di Sony (2003) e HRP-4C del giapponese National Institute of Advanced Industrial Science and Technology (2009). Questi robot son caratterizzati da una camminata statica, nella quale il centro di massa e' sempre mantenuto all'interno della superficie di appoggio del robot.

Sempre nel campo dei robot antropomorfi sono stati fatti grandi sforzi per studiare e riprodurre la camminata dinamica dell'uomo, permettendo movimenti piu' fluidi e naturali. Alcuni ricercatori hanno sfruttato particolari costruzioni meccaniche per realizzare una camminata dinamica passiva[1], che minimizza il consumo di energia necessario per gli spostamenti[2]. Al-



(a) (b)

Figura 2.1: Robot antropomorfi a camminata statica: (a) Honda Asimo, 2000 (b) HRP-4C, 2009

tri ricercatori hanno inseguito l'approccio attivo, costruendo robot antropomorfi che si mantengono costantemente in equilibrio e sono in grado di camminare, saltare e correre (Dexter, Anybots[3]). La naturale attitudine di questo tipo di robot a lavorare in ambienti costruiti per gli essere umani è però contrastata dalla difficoltà di realizzazione, dovuta soprattutto alla complessità meccanica, alla difficoltà nel controllo e agli elevati consumi energetici.

Una soluzione alternativa ai robot antropomorfi per costruire piattaforme in grado di operare in una buona parte degli ambienti destinati all'uomo consiste nel realizzare robot bilancianti, dotati di due ruote motrici coassiali, che mantengono la struttura in costante equilibrio, unendo i vantaggi del movimento su ruote alla ridotta impronta a terra che permette movimenti agili in ambienti domestici. In questo modo si possono realizzare robot capaci di spostare carichi relativamente pesanti con richieste energetiche contenute, adatti a muoversi in ambienti domestici e in grado di mantenere il corpo ad un'altezza elevata per una migliore interazione con l'uomo. Soluzioni di questo tipo sono state adottate anche per prodotti commerciali, come la piattaforma mobile Segway[4], che ha riscosso grande successo sul mercato,

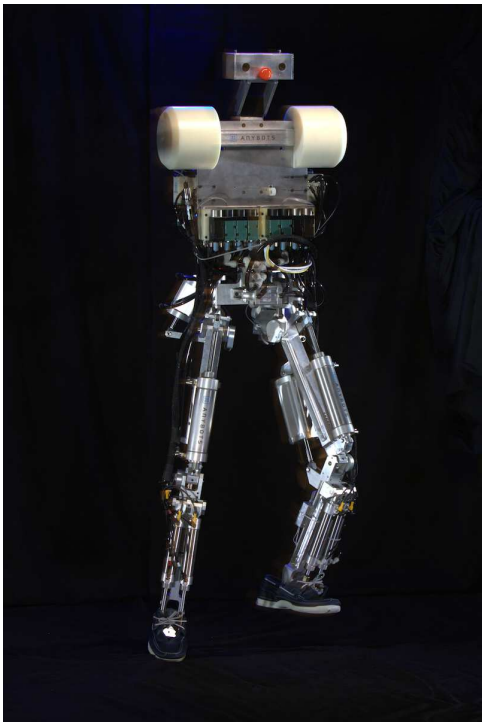


Figura 2.2: Anybots Dexter: robot antropomorfo a camminata dimanica

o il robot per la telepresenza QA, sempre sviluppato presso Anybots.

Il controllo di un robot bilanciante si riconduce al mantenimento dell'equilibrio di un pendolo inverso, uno tra i sistemi più studiati in letteratura per via della sua naturale instabilità. Questa caratteristica oltre alla possibilità di una formulazione relativamente semplice (o comunque semplificabile) delle relazioni analitiche tra le variabili che descrivono il sistema, hanno portato gli studiosi dei sistemi di controllo ad utilizzare la stabilizzazione del pendolo inverso come applicazione per lo studio e l'analisi dei sistemi di controllo e come metro di paragone delle prestazioni tra differenti controllori. I primi ad utilizzare il pendolo inverso come strumento di benchmark per sistemi di controllo sono stati, nel 1968, Michie e Chambers[5], ma la più grande popolarità è stata raggiunta dopo la pubblicazione, nel 1983, di [6] da parte di A.G. Barto e R.S. Sutton, i padri dell'apprendimento per rinforzo. Gli algoritmi proposti per risolvere il problema del pendolo inverso spaziano dall'implementazione di tecniche appartenenti alla teoria del controllo classico ad approcci più recenti basati su reti neurali e sulla teoria dell'apprendimento per rinforzo.

(a) (b)

Figura 2.3: Robot bilancianti commerciali: (a) Segway, 2002 (b) Anybots QA, 2009

Nell'ambito del controllo classico, sono stati proposti ed analizzati svariati controllori in grado di mantenere l'equilibrio instabile del pendolo inverso.

Numerosi robot costruiti a scopo didattico, tra i quali i più noti sono quelli costruiti da BalBots[7], implementano un controllore PID utilizzato per il mantenimento dell'equilibrio, alla cui azione viene aggiunto un contributo proporzionale alla velocità di spostamento che si vuole raggiungere. Questo approccio fa si che il movimento del robot sia accompagnato da una continua oscillazione attorno al punto di equilibrio, a causa del disturbo che viene introdotto sull'azione prodotta dal controllore. Un comportamento di questo tipo può produrre effetti tollerabili su un robot di piccole dimensioni, ma non è adatto quando le dimensioni e le masse in gioco diventano significative.
Un altro metodo per far inseguire un profilo di velocità ad un sistema mantenuto in equilibrio tramite un controllore PID consiste nell'aggiungere un piccolo offset all'angolo inseguito, così che il robot debba accellerare in una direzione per mantenere il setpoint. Questa soluzione, adottata da nBot di David P. Anderson[8], è di nuovo adatta soltanto a robot di piccole dimensioni: se il baricentro del robot è posto ad un'altezza significativa, come nel

(a) (b)

Figura 2.4: Robot bilancianti realizzati per lo studio di controllori classici: (a) nBot, 2001 (b) Balbot, 2005

caso in esame, un setpoint molto piccolo, paragonabile con la risoluzione del sensore, porta a velocità di movimento troppo elevate per essere controllate.

Altri gruppi di ricerca hanno affrontato il problema applicando la teoria del controllo ottimo[9][10], sintetizzando un controllore LQR che riceve in ingresso tutti gli stati del sistema (angolo, velocità angolare, posizione e velocità lineare) e produce un'azione che minimizza una determinata funzione di costo. Anche la piattaforma commerciale Segway è governata da un controllore di questo tipo. La taratura del controllore LQR non può essere però effettuata con regole pratiche come nel caso dei PID, ed è quindi necessario definirne i parametri mediante simulazioni. Sarà quindi necessario,

nella realizzazione del robot, progettare un sistema di controllo dei motori che sia in grado di inseguire con alta precisione un setpoint in coppia, per rispettare le condizioni simulate.

Sulla base delle tipologie di controllo classiche sono state sviluppati anche forme di controllo ibride, ad esempio sfruttando la logica fuzzy per la taratura dei parametri di un controllore PID[11], oppure introducendo delle reti neurali per stimare i comportamenti non lineari del sistema[12].

Il problema del bilanciamento di un pendolo inverso è stato approfonditamente studiato anche nel campo dell'apprendimento automatico. Oltre ai già citati A.G. Barto e R.S. Sutton, che hanno utilizzato proprio questo problema per valutare l'efficacia delle tecniche da loro presentate, molti altri ricercatori hanno avuto successo nel mantenere in equilibrio una struttura a pendolo inverso utilizzando soluzioni completamente differenti rispetto a quelle mostrate fino ad ora.
Sono stati presentati studi in cui una rete neurale viene dapprima utilizzata per prevedere il comportamento dinamico del sistema, senza basarsi su un modello, e quindi espansa al fine di imparare una strategia di controllo in grado di far seguire alle variabili di stato delle traiettorie predefinite[13].
Altri approcci *model-free* sono stati proposti sfruttando il paradigma dell'apprendimento per rinforzo[14], in cui l'algoritmo di controllo viene addestrato osservando una serie di dati raccolti mentre il robot esegue azioni completamente casuali. Sulla base dello spazio visitato, l'algoritmo riesce ad apprendere la dinamica del moto in relazione all'azione compiuta e a generalizzare la conoscenza acquisita per estenderla anche a stati mai visitati, ricavando un modello del sistema. Viene quindi stimata una politica di controllo in grado di mantenere l'equilibrio del robot ed inseguire un determinato setpoint. Questa tecnica ha mostrato ottimi risultati al simulatore, rendendo interessante una sua applicazione ad un robot reale.

In questo lavoro di tesi ci si propone di realizzare una piattaforma mobile bilanciante di dimensioni tali da rendere possibile il movimento in ambienti domestici e l'interazione con l'uomo, in grado di trasportare eventuali dispositivi accessori e capace di seguire un profilo di velocità. Dopo aver progettato e realizzato l'hardware, ci si concentrerà sulla strategia di controllo, analizzando sia gli approcci classici, quali controllori PID e LQR, sia tecniche innovative basate sull'apprendimento per rinforzo, che hanno mostrato risultati promettenti nell'ambiente simulato.

Capitolo 3

Studio teorico del problema

"Se la conoscenza può creare dei problemi, non è con l'ignoranza che possiamo risolverli."

Isaac Asimov

L'obiettivo della tesi, dal punto di vista concettuale, può essere raggiunto risolvendo due problemi distinti:

- il mantenimento dell'equilibrio del robot, ovvero la sintesi di un controllore che sia in grado, conoscendo lo stato del robot, di scegliere le azioni da eseguire affinchè il robot non cada
- la stima dell'angolo del telaio del robot rispetto alla verticale, utilizzando i dati messi a disposizione dai sensori presenti che, sebbene producano misure legate all'angolo, non ne forniscono direttamente il valore

Per affrontare il primo problema è necessario descrivere il sistema mediante relazioni matematiche, al fine di ricavare un modello utilizzabile in un ambiente di simulazione in cui progettare il controllore.
La stima dell'angolo mediante i sensori montati sul robot richiede invece la realizzazione di un filtro in grado di estrarre l'informazione utile da dati disturbati da diverse forme di rumore.
Nelle sezioni seguenti vengono riportati i concetti alla base delle scelte adottate per la soluzione dei problemi descritti.

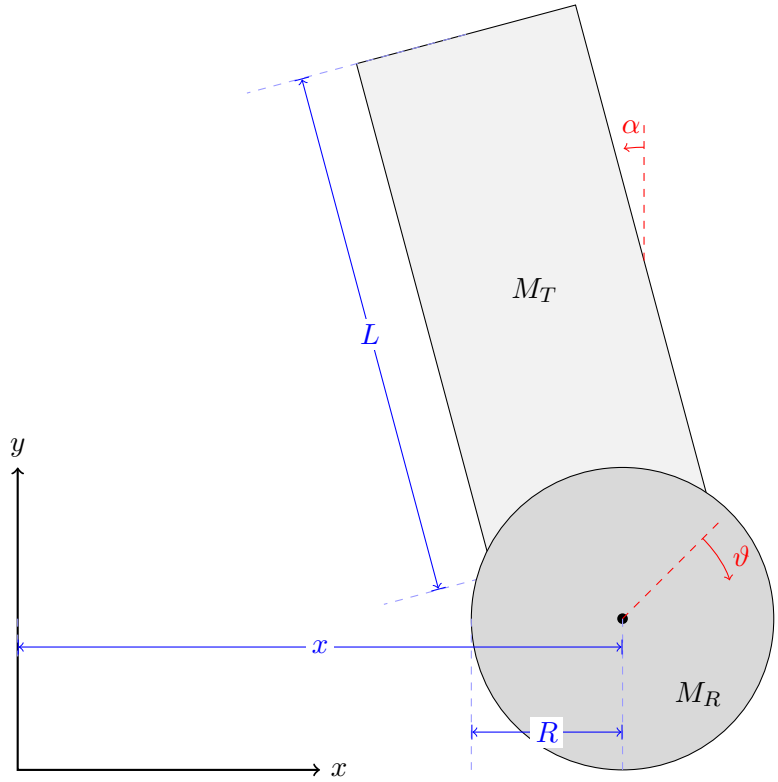


Figura 3.1: Modello semplificato del robot utilizzato per ricavare la descrizione della dinamica del sistema

3.1 Modello del robot

3.1.1 Modello dinamico

Il modello dinamico del robot è stato ricavato riducendo il sistema allo schema mostrato in Figura 3.1. In Tabella 3.1 sono riportati i simboli utilizzati.

Per descrivere il sistema dal punto di vista dinamico si possono scomporre le forze in gioco nelle componenti orizzontali ed orizzontali. Chiamando N la componente orizzontale della forza che il pendolo, ovvero il telaio, esercita sul carrello, ovvero sull'asse a cui le ruote ruote sono fissate, e P la componente verticale della stessa forza, si può procedere al bilanciamento delle due componenti.

Tabella 3.1: Descrizione dei simboli utilizzati

R	Raggio delle ruote
L	Altezza del telaio
M_R	Massa di una ruota
M_T	Massa del telaio
x	Distanza dell'asse di rotazione dall'origine
α	Angolo tra il telaio e la verticale
ϑ	Angolo di rotazione delle ruote
N	Componente orizzontale delle forze agenti sull'asse di rotazione delle ruote
P	Componente verticale delle forze agenti sull'asse di rotazione delle ruote

Le forze orizzontali agenti sull'asse delle ruote sono:

$$F = M_R\ddot{x} + b\dot{x} + N \tag{3.1}$$

in cui $b\dot{x}$ rappresenta l'attrito dinamico del robot.
Le forze orizzontali agenti sul telaio sono:

$$N = M_T\ddot{x} - M_T\frac{L}{2}\dot{\alpha}^2 sin\alpha + M_T\frac{L}{2}\ddot{\alpha}cos\alpha \tag{3.2}$$

in cui $\frac{L}{2}\dot{\alpha}^2 sin\alpha$ rappresenta la componente normale dell'accelerazione del baricentro del telaio e $\frac{L}{2}\ddot{\alpha}cos\alpha$ rappresenta la componente tangenziale.
Sostituendo in 3.1 l'espressione di N trovata in 3.2 si ricava la prima equazione del moto:

$$F = M_R\ddot{x} + b\dot{x} + M_T\ddot{x} - M_T\frac{L}{2}\dot{\alpha}^2 sin\alpha + M_T\frac{L}{2}\ddot{\alpha}cos\alpha \tag{3.3}$$

La seconda equazione del moto si ottiene sommando le componenti verticali delle forze:

$$Psin\alpha + Ncos\alpha = M_T g sin\alpha + M_T\frac{L}{2}\ddot{\alpha} + M_T\ddot{x}cos\alpha \tag{3.4}$$

Bilanciando i momenti agenti sul telaio si può scrivere:

$$\frac{L}{2}Ncos\alpha + \frac{L}{2}Psin\alpha + I_T\ddot{\alpha} = 0 \tag{3.5}$$

in cui I_T rappresenta il momento di inerzia del telaio, che verrà trattato dettagliamente nella sezione successiva.
Mettendo a sistema 3.4 e 3.5 si ricava la seconda equazione del moto:

$$(I_T + M_T\frac{L^2}{4})\ddot{\alpha} + M_T g\frac{L}{2}sin\alpha + M_T\frac{L}{2}\ddot{x}cos\alpha = 0 \tag{3.6}$$

Il moto del modello è completamente descritto dal sistema di equazioni 3.3 e 3.6, in cui si è introdotta la coppia C esercitata dai motori sulle ruote, sostituendo quindi $F = C/R$:

$$\begin{cases} M_R\ddot{x} + b\dot{x} + M_T\ddot{x} - M_T\frac{L}{2}\dot{\alpha}^2 sin\alpha + M_T\frac{L}{2}\ddot{\alpha}cos\alpha = C/R \\ (I_T + M_T\frac{L^2}{4})\ddot{\alpha} + M_T g\frac{L}{2}sin\alpha + M_T\frac{L}{2}\ddot{x}cos\alpha = 0 \end{cases} \tag{3.7}$$

3.1.2 Soluzione del sistema mediante equazioni di Lagrange

Tra le varie strade percorribili per esplicitare il moto del sistema si è scelto di ricorrere al metodo delle equazioni di Lagrange, che prevede di risolvere un sistema di n equazioni la cui forma concisa è:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_k} - \frac{\partial T}{\partial q_k} = Q_k \qquad k = 1, \ldots, n \tag{3.8}$$

in cui T rappresenta l'energia cinetica totale del sistema, q_k sono le *coordinate lagrangiane* (una per ogni grado di libertà) che descrivono il sistema e Q_k sono delle quantità scalari denominate *componenti generalizzate di Lagrange*. Nel caso in esame si possono scegliere come coordinate lagrangiane l'angolo ϑ che indica la rotazione delle ruote e l'angolo α compreso tra il telaio e la direzione dell'accelerazione di gravità.
Le componenti generalizzate di Lagrange Q_k sono definite come:

$$Q_k \equiv \sum_{i=1}^{N} (\mathbf{F_i} + \mathbf{f_i}) \times \frac{\partial \mathbf{P_i}}{\partial \mathbf{q_k}} \tag{3.9}$$

in cui F_i e f_i rappresentano le forze esterne ed interne e P_i lo spostamento virtuale (dal *principio dei lavori virtuali*). In sostanza, il valore di Q_k indica il contributo della risultante di tutte le forze in gioco alla quantità di lavoro virtuale nella direzione della coordinata q_k. La soluzione delle k equazioni di Lagrange permette di descrivere completamente la dinamica del sistema.

Nel caso in cui siano presenti contributi conservativi (come l'energia potenziale gravitazionale) e contributi non conservativi (nel caso in esame si tratta della coppia C esercitata dai motori), si può ricorrere alla formulazione *mista* delle equazioni di Lagrange, che tiene conto di entrambi i contributi:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_k} - \frac{\partial T}{\partial q_k} = \frac{\partial U}{\partial q} + Q'_k \qquad k = 1, \ldots, n \tag{3.10}$$

in cui U è un campo conservativo e Q'_k sono i contributi delle forze non conservative.

La prima grandezza cinematica da ricavare per poter risolvere il sistema di equazioni di Lagrange è l'energia cinetica T, che per un punto materiale sottoposto ad un atto di moto traslatorio è definita come:

$$T = \frac{1}{2}mv^2 \tag{3.11}$$

mentre nel caso di moto rotatorio si introduce il momento di inerzia I, che indica la distribuzione delle masse di un corpo rispetto ad un asse (in questo caso l'asse della rotazione ω), distante r dal centro di massa, definito per il punto materiale come:

$$I_\omega = mr^2 \tag{3.12}$$

e l'energia cinetica T assume la forma:

$$T = \frac{1}{2}I_\omega\omega^2 \tag{3.13}$$

L'energia cinetica di un sistema materiale qualunque può essere espressa ricorrendo al *Teorema di Konig*:

$$T = \frac{1}{2}m\bar{v}^2 + \frac{1}{2}\bar{I}\omega^2 \tag{3.14}$$

L'energia del sistema è quindi pari alla somma dell'energia cinetica che avrebbe il baricentro, se in esso fosse concentrata tutta la massa del sistema, e dell'energia cinetica del sistema materiale nel suo atto di moto relativo al proprio baricentro.

Si procede quindi con il calcolo delle componenti dell'energia cinetica dovute al moto traslatorio e al moto rotatorio delle varie componenti del sistema.

Per quanto riguarda le ruote, si valuta dapprima il valore del momento d'inerzia, estendendo la definizione 3.12, valida per il punto materiale, al corpo rigido.
Se si considera un corpo rigido di volume V come un insieme di punti, ciascuno caratterizzato da un volume δV e da una massa $\delta m = \rho\delta V$, dove ρ è la densità, il contributo di ogni punto al momento di inerzia del corpo e‘ dato da

$$\delta I_\omega = \rho\delta V r^2 \tag{3.15}$$

in cui r indica la distanza del punto dall'asse di rotazione.
Il momento di inerzia del corpo si ottiene allora sommando tutti i contribuiti e passando al continuo, cioè imponendo $V \to 0$:

$$I_\omega = \int_V \rho r^2 dV \tag{3.16}$$

Nel calcolo del momento di inerzia della ruota, essa può essere considerata un corpo *omogeneo* (la densità ρ è funzione costante), di raggio R e altezza H, ottenendo:

$$I_R = \int_0^R \rho r^2 H 2\pi r dr = 2\pi\rho H \int_0^R r^3 dr = \frac{\pi\rho H R^4}{2} = \frac{1}{2}MR^2 \tag{3.17}$$

Il contributo all'energia cinetica del sistema dovuto a ciascuna ruota, considerando la componente dovuta al moto traslatorio e quella dovuta al moto rotatorio, vale quindi:

$$\begin{aligned} T_R &= \frac{1}{2}m\bar{v}^2 + \frac{1}{2}\bar{I}\omega^2 \\ &= \frac{1}{2}M_R\dot{x}_C^2 + \frac{1}{2}(\frac{1}{2}MR^2)\dot{\vartheta}^2 \\ &= \frac{1}{2}M_R\dot{x}_C^2 + \frac{1}{4}MR^2\dot{\vartheta}^2 \end{aligned} \tag{3.18}$$

Assumendo vincolo di puro rotolamento, ovvero $\dot{x}_C = R\dot{\vartheta}$, la quantità di energia cinetica legata al moto della ruota può essere scritta come:

$$\begin{aligned} T_R &= \frac{1}{2}M_R(R\dot{\vartheta})^2 + \frac{1}{4}MR^2\dot{\vartheta}^2 \\ &= \frac{3}{4}M_R(R\dot{\vartheta})^2 \end{aligned} \tag{3.19}$$

L'energia cinetica connessa al moto del telaio può essere calcolata in modo analogo, utilizzando nuovamente il teorema di Konig enunciato nell'equazione 3.14[15].
È innanzitutto necessario analizzare il moto del telaio ricorrendo allo stesso sistema di riferimento utilizzato in precedenza, esplicitando la distanza tra il centro di massa del telaio G e l'origine del sistema di riferimento O:

$$(G-O) = (x_C + \frac{L}{2}sin\alpha)\hat{i} + (R + \frac{L}{2}cos\alpha)\hat{j} \tag{3.20}$$

La velocità del centro di massa $\bar{v}_G$ e il suo quadrato, utili nel calcolo dei contributi all'energia cinetica dovuti al moto traslatorio e rotatorio, valgono:

$$\bar{v}_G = \frac{\partial}{\partial t}(G-O) = (R\dot{\theta} + \frac{L}{2}\dot{\alpha}cos\alpha)\hat{i} - \frac{L}{2}\dot{\alpha}sin\alpha)\hat{j} \tag{3.21}$$

$$\bar{v}_G^2 = \frac{\partial^2}{\partial t^2}(G-O) = R^2\dot{\theta}^2 + \frac{L^2}{4}\dot{\alpha}^2 + R\dot{\vartheta}L\dot{\alpha}cos\alpha \tag{3.22}$$

Il momento d'inerzia del telaio viene calcolato assumendo nuovamente che il corpo sia omogeneo:

$$I_T = \int_0^L \rho r^2 dr = \rho \int_0^L r^2 dr = \rho\frac{1}{3}L^3 = \frac{1}{3}M_T L^2 \tag{3.23}$$

L'energia cinetica legata al moto del telaio vale quindi:

$$\begin{aligned} T_T &= \frac{1}{2}m\bar{v}^2 + \frac{1}{2}\bar{I}\omega^2 \\ &= \frac{1}{2}M_T(R^2\dot{\theta}^2 + \frac{L^2}{4}\dot{\alpha}^2 + R\dot{\vartheta}L\dot{\alpha}cos\alpha) + \frac{1}{2}(\frac{1}{3}M_TL^2)\dot{\alpha}^2 \\ &= \frac{1}{2}M_T(R^2\dot{\theta}^2 + \frac{L^2}{4}\dot{\alpha}^2 + R\dot{\vartheta}L\dot{\alpha}cos\alpha) + \frac{1}{6}M_TL^2\dot{\alpha}^2 \end{aligned} \tag{3.24}$$

Unendo i risultati trovati in 3.18 e in 3.24 si può scrivere l'energia cinetica totale del sistema:

$$\begin{aligned} T &= 2T_R + T_T \\ &= \frac{3}{2}M_R(R\dot{\vartheta})^2 \\ &\quad + \frac{1}{2}M_T(R^2\dot{\theta}^2 + \frac{L^2}{4}\dot{\alpha}^2 + R\dot{\vartheta}L\dot{\alpha}cos\alpha) + \frac{1}{6}M_TL^2\dot{\alpha}^2 \end{aligned} \tag{3.25}$$

Determinata l'espressione dell'energia cinetica totale del sistema, si può procedere con l'esplicitare il valore dell'energia potenziale:

$$U = -M_Tg(R + \frac{L}{2}cos\alpha) \tag{3.26}$$

La variazione dell'energia potenziale U dipende solo dall'angolo α tra il telaio e la direzione dell'accelerazione di gravità:

$$\frac{\partial U}{\partial \vartheta} = 0 \tag{3.27}$$

$$\frac{\partial U}{\partial \alpha} = M_Tg\frac{L}{2}\dot{\alpha}sin\alpha \tag{3.28}$$

Le ultime grandezze da valutare sono i contributi non conservativi Q', che possono essere determinati ricorrendo al principio dei lavori virtuali:

$$\partial\mathcal{L} = C\partial\vartheta - C\partial\alpha \tag{3.29}$$

È quindi possibile isolare le componenti generalizzate di Lagrange relative alle coordinate lagrangiane ϑ e α:

$$Q'_\vartheta = C \tag{3.30}$$

$$Q'_\alpha = -C \tag{3.31}$$

Si può ora procedere con la scrittura delle equazioni di Lagrange, sostituendo nella 3.10 le rispettive espressioni. Per quanto riguarda la coordinata

lagrangiana ϑ si ha:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\vartheta}} - \frac{\partial T}{\partial \vartheta} = \frac{\partial U}{\partial \vartheta} + Q'_{\vartheta} \tag{3.32}$$

Come visto in precedenza, T non dipende da ϑ quindi $\frac{\partial T}{\partial \vartheta} = 0$, ed allo stesso modo U non dipende da ϑ, quindi $\frac{\partial U}{\partial \vartheta} = 0$.
Resta da calcolare il valore di $\frac{d}{dt}\frac{\partial T}{\partial \dot{\vartheta}}$:

$$\frac{\partial T}{\partial \dot{\vartheta}} = (3M_R + M_T)R^2\dot{\vartheta} + \frac{1}{2}M_T R\dot{\vartheta}L\dot{\alpha}cos\alpha \tag{3.33}$$

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\vartheta}} = (3M_R + M_T)R^2\ddot{\vartheta} + \frac{1}{2}M_T RL(\ddot{\alpha}cos\alpha - \dot{\alpha}^2 sin\alpha) \tag{3.34}$$

$$\tag{3.35}$$

Sostituendo nella 3.32 le espressioni ricavate si può infine scrivere la prima equazione di Lagrange:

$$(3M_R + M_T)R^2\ddot{\vartheta} + \frac{1}{2}M_T RL(\ddot{\alpha}cos\alpha - \dot{\alpha}^2 sin\alpha) = C \tag{3.36}$$

Per quanto riguarda la coordinata lagrangiana α si ha:

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\alpha}} - \frac{\partial T}{\partial \alpha} = \frac{\partial U}{\partial \alpha} + Q'_{\alpha} \tag{3.37}$$

In questo caso sia nell'espressione dell'energia potenziale U appare la variabile α, mentre T non mostra dipendenza da α, ovvero $\frac{\partial T}{\partial \alpha} = 0$. Si determinano quindi i valori di $\frac{d}{dt}\frac{\partial T}{\partial \dot{\alpha}}$ e di $\frac{\partial U}{\partial \alpha}$:

$$\frac{\partial T}{\partial \dot{\alpha}} = \frac{1}{4}M_T L^2\dot{\alpha} + \frac{1}{2}M_T R\dot{\vartheta}Lcos\alpha + \frac{1}{3}M_T L^2\dot{\alpha} \tag{3.38}$$

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{\alpha}} = \frac{1}{4}M_T L^2\ddot{\alpha} + \frac{1}{2}M_T RL(\ddot{\vartheta}cos\alpha - \dot{\vartheta}\dot{\alpha}sin\alpha) + \frac{1}{3}M_T L^2\ddot{\alpha} \tag{3.39}$$

$$\frac{\partial U}{\partial \alpha} = M_T g\frac{L}{2}sin\alpha \tag{3.40}$$

Sostituendo nella 3.37 le espressioni ricavate si può infine scrivere la seconda equazione di Lagrange:

$$\frac{1}{4}M_T L^2\ddot{\alpha} + \frac{1}{2}M_T RL(\ddot{\vartheta}cos\alpha - \dot{\vartheta}\dot{\alpha}sin\alpha) + \frac{1}{3}M_T L^2\ddot{\alpha} = M_T g\frac{L}{2}sin\alpha - C \tag{3.41}$$

Il moto del sistema in funzione delle forze agenti su di esso è quindi descritto dalle due equazioni ricavate:

$$\begin{cases} (3M_R + M_T)R^2\ddot{\vartheta} + \frac{1}{2}M_T RL(\ddot{\alpha}cos\alpha - \dot{\alpha}^2 sin\alpha) = C \\ \frac{1}{4}M_T L^2\ddot{\alpha} + \frac{1}{2}M_T RL(\ddot{\vartheta}cos\alpha - \dot{\vartheta}\dot{\alpha}sin\alpha) + \frac{1}{3}M_T L^2\ddot{\alpha} = M_T g\frac{L}{2}sin\alpha - C \end{cases} \tag{3.42}$$

3.1.3 Simulazione del modello in ambiente MATLAB/Simulink

Tramite il pacchetto Simulink per MATLAB è possibile simulare sistemi non lineari inserendo sotto forma di schemi a blocchi le equazioni che descrivono il modello.
Si è scelto quindi di utilizzare questo software per simulare il sistema governato dalle equazioni 3.7, qui riportate per comodità:

$$\begin{cases} M_R\ddot{x} + b\dot{x} + M_T\ddot{x} - M_T\frac{L}{2}\dot{\alpha}^2 sin\alpha + M_T\frac{L}{2}\ddot{\alpha}cos\alpha = \frac{C}{R} \\ (I_T + M_T\frac{L^2}{4})\ddot{\alpha} + M_T g\frac{L}{2}sin\alpha + M_T\frac{L}{2}\ddot{x}cos\alpha = 0 \end{cases}$$

Lo schema realizzato è mostrato in Figura 3.2.

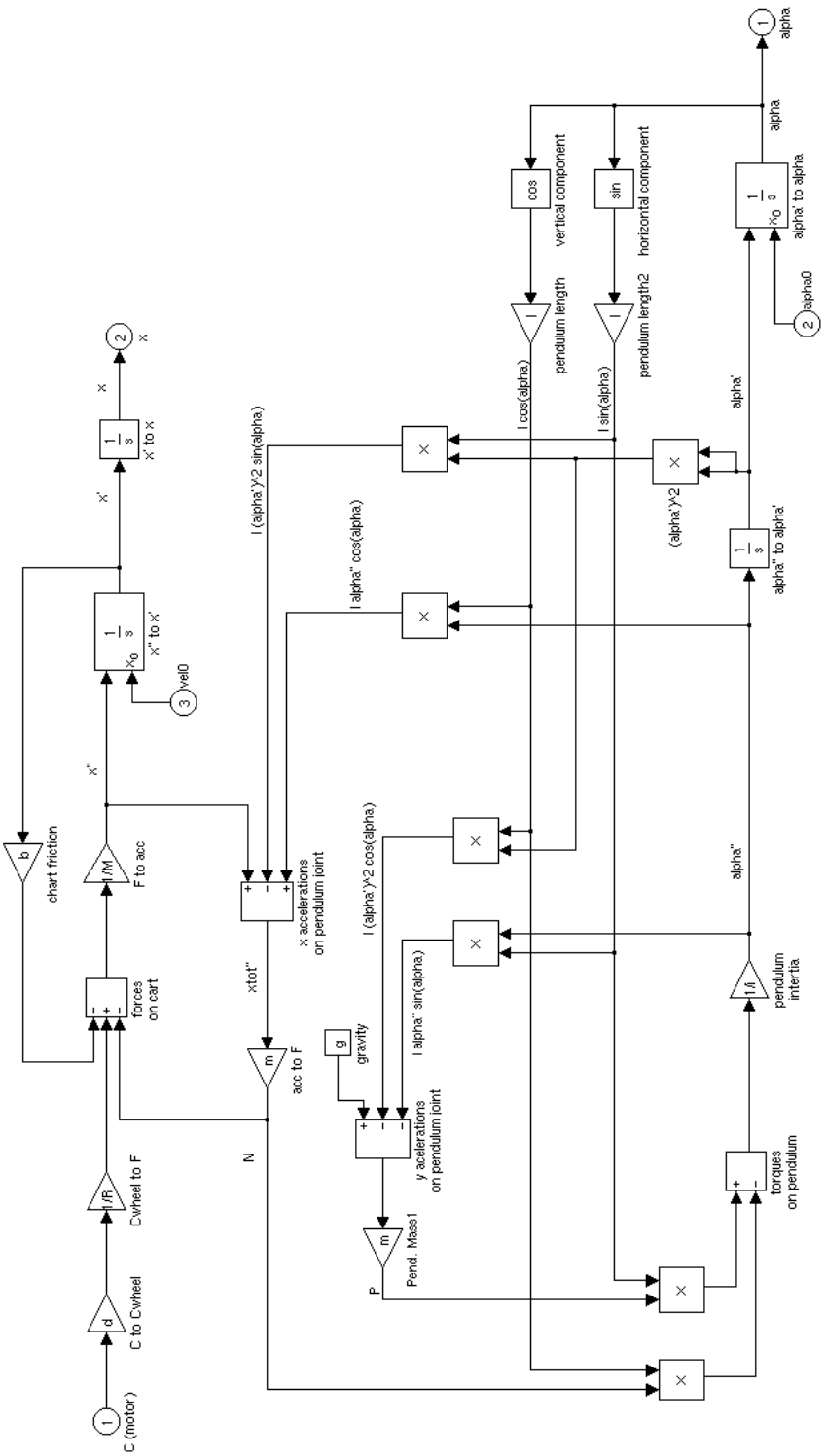


Figura 3.2: Modello del robot riportato in ambiente MATLAB/Simulink

La parte superiore rappresenta principalmente la prima equazione del moto, ovvero quella che descrive la componente orizzontale delle forze agenti sull'asse delle ruote. É possibile impostare come ingresso la coppia C esercitata dai motori, l'angolo e la velocita' iniziale del robot, indicati rispettivamente con $alpha0$ e $vel0$.
Nei blocchi inferiori si possono invece riconoscere i fattori della seconda equazione del moto, in cui compaiono i momenti di inerzia.
Si nota come in ambiente Simulink si possano descrivere sistemi non lineari con grande facilità, permettendo di analizzare con MATLAB le equazioni ottenute mediante lo studio della dinamica del sistema.

Il modello del pendolo inverso è stato quindi inserito in un anello di controllo, mostrato in Figura 3.3, in modo da poter simulare il comportamento dei differenti controllori classici utilizzati, descritti nel Capitolo 7. Lo stesso modello è stato anche utilizzato per il dimensionamento degli attuatori, secondo la procedura descritta nella Sezione 4.3.

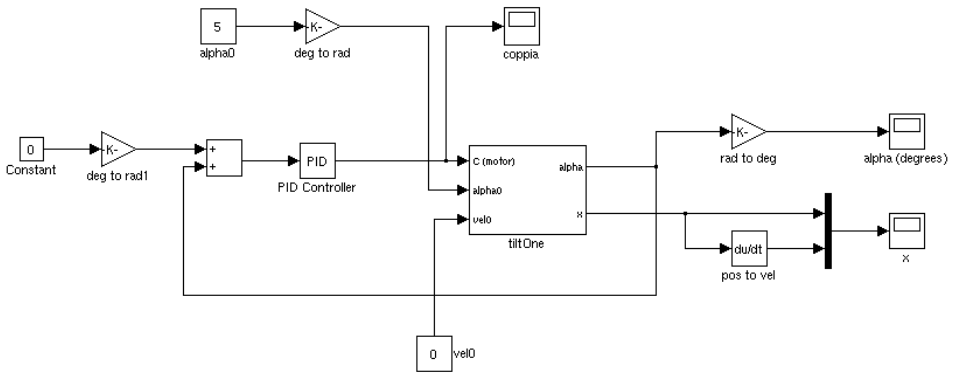


Figura 3.3: Anello di controllo in ambiente MATLAB/Simulink

3.2 Stima dell'inclinazione del robot

3.2.1 Filtro di Kalman

Il filtro di Kalman è un filtro ricorsivo che stima lo stato di un sistema dinamico sulla base dello stato precedente e di una serie di misure soggette a rumore. Il termine ricorsivo indica che è sufficiente conoscere la stima precedente e la misura corrente per calcolare una stima dello stato attuale.

Il filtro di Kalman permette di valutare lo stato x di un sistema descritto da un'equazione differenziale lineare del tipo:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{3.43}$$

mediante una misura z così definita:

$$z_k = Hx_k + v_k \tag{3.44}$$

in cui:

- $\mathbf{x_k}$ è il vettore di stato del sistema all'istante k
- $\mathbf{A}$ è la matrice di stato, che descrive l'evoluzione libera della variabile di stato rispetto al suo valore attuale
- $\mathbf{B}$ è la matrice degli ingressi, che descrive l'evoluzione forzata della variabile di stato rispetto al valore attuale dell'ingresso
- $\mathbf{u_{k-1}}$ è il vettore degli ingressi che hanno agito sul sistema all'istante $k-1$
- $\mathbf{w_{k-1}}$ rappresenta i disturbi che hanno influenzato lo stato del sistema all'istante $k-1$
- $\mathbf{z_k}$ è il vettore delle misure dello stato del sistema all'istante k
- $\mathbf{H}$ è la matrice delle uscite, che descrive il valore assunto dalle variabili misurate in funzione del valore attuale dello stato del sistema
- $\mathbf{v_k}$ rappresenta i disturbi che influenzano la misura dello stato del sistema all'istante $k-1$

Le variabili casuali w_k e v_k rappresentano rispettivamente il rumore del processo e il rumore della misura. Si assume che esse siano tra loro scorrelate ed affette da rumore bianco con distribuzione normale:

$$p(w) \sim N(0, Q) \tag{3.45}$$

$$p(v) \sim N(0, R) \tag{3.46}$$

in cui Q indica la covarianza del rumore di processo e R la covarianza del rumore di misura.
Il filtro è caratterizzato da due fasi distinte (Figura 3.4): la fase di *predict*, in cui sulla base dell'ultima stima x_{k-1} dello stato del sistema, senza conoscere la misura z, viene stimato lo stato successivo x_k, e la fase di *update*, in cui

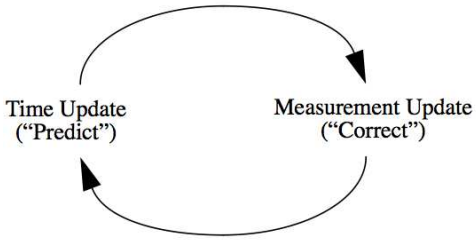


Figura 3.4: Ciclo del filtro di Kalman

la stima x_k effettuata a priori viene corretta mediante la misura attuale z_k.

Nella fase di predict vengono effettute due operazioni:

- si esegue la stima a priori $\hat{x}_k^-$ dello stato del sistema, conoscendo la stima precedente $\hat{x}_{k-1}$ e l'uscita u_{k-1}

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1} \tag{3.47}$$

- si aggiorna la stima della matrice di covarianza dell'errore P_k^- conoscendo la stima precedente P_{k-1} e la covarianza del rumore di processo Q

$$P_k^- = AP_{k-1}A^T + Q \tag{3.48}$$

Durante la fase di update si eseguono invece i seguenti passi:

- si valuta il guadagno di Kalman K, tramite cui vengono definiti il peso della stima e il peso della misura nel valutare lo stato del sistema

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1} \tag{3.49}$$

- si corregge la stima a priori $\hat{x}_k^-$ pesando opportunamente la stima effettuata e la differenza tra la misura effettiva z_k e la stima della misura $H\hat{x}_k^-$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-) \tag{3.50}$$

- si corregge la stima della matrice di covarianza dell'errore P_k^-

$$P_k = (I - K_k H)P_k^- \tag{3.51}$$

Il guadagno di Kalman K, valutato nell'Equazione 3.49, può essere interpretato in questo modo: se la covarianza dell'errore di misura R tende a zero, la misura z_k viene pesata maggiormente, mentre la misura stimata $H\hat{x}_k^-$ ha meno rilievo; viceversa, se è la stima a priori della matrice di covarianza dell'errore P_k^- a tendere a zero, la stima della misura ha un peso maggiore a scapito di z_k.

3.2.2 Stima dell'angolo mediante filtro di Kalman

A bordo del robot sono presenti due sensori utilizzati per la stima dell'angolo del telaio rispetto alla verticale: un accelerometro, tramite cui viene misurata l'accelerazione di gravità scomposta nelle componenti parallele ed ortogonali al telaio, e un giroscopio, che fornisce la velocità angolare a cui il robot sta ruotando attorno all'asse delle ruote.
Sebbene possa sembrare che questi sensori siano ridondanti, in quanto entrambi possano essere usati per ricavare una misura dell'angolo, ci sono in realtà problemi di natura diversa nel loro utilizzo:

- tramite i dati forniti dall'accelerometro possono essere misurate con precisione le componenti ortogonali dell'accelerazione di gravità, dalle quali è facile ricavare l'inclinazione del telaio rispetto ad essa, ma il sensore percepisce anche tutte le altre accelerazioni a cui il robot è sottoposto, dovute ad esempio al suo stesso movimento, che vanno a sovrapporsi al dato di interesse
- la velocità angolare rilevata dal giroscopio può essere integrata nel tempo per valutare il valore dell'angolo, ma il rumore che affligge il sensore porta ad avere un *drift* della misura ricavata: un offset anche ridotto, ma continuato nel tempo, fa si che l'integrale diverga portando ad una misura il cui errore continua a crescere

Il filtro di Kalman descritto nella sezione precedente si presta ad effettuare una stima dell'angolo, effettuando le operazioni di predict e di update utilizzando i dati forniti dai due sensori:

fase di predict : basandosi sulla velocità angolare fornita dal giroscopio, che indica come lo stato del sistema sta evolvendo, il filtro stima l'angolo che il robot andrà ad assumere; alla misura fornita dal sensore

viene sottratta la stima dell'entità del'offset, calcolata durante l'update precedente

fase di update : utilizzando la misura dell'angolo ricavata dalle letture dell'accelerometro, la stima dell'angolo viene corretta e si aggiorna il valore dell'offset che affligge il giroscopio

Si rimanda alla Sezione 8.1, in cui sono mostrati i risultati delle prove sperimentali, per osservare l'efficacia del filtro nella stima dell'angolo del robot.

Capitolo 4

Progetto meccanico

"Bender: Ma sei sicuro di volere un Robot per amico?
Fry: Certo, da quando avevo 6 anni! "

Futurama

Il progetto meccanico riguarda tutti gli elementi che costituiscono il robot, partendo dalla scelta della disposizione delle varie componenti e dal dimensionamento del telaio per arrivare alla realizzazione di ruote apposite e alla scelta degli attuatori. In seguito sono descritte le scelte progettuali effettuate a riguardo dei vari elementi che compongono il robot, di cui è riportata una fotografia in Figura 4.1.

4.1 Telaio

I requisiti principali per la realizzazione del telaio, frutto dell'esperienza nel progetto di piattaforme simili in passato, sono la semplicità di assemblaggio e disassemblaggio, la facilità di accedere alle parti interne e la possibilità di aggiungere e modificare elementi in futuro. Per soddisfare queste richieste si è scelto di realizzare il telaio utilizzando dei profilati di alluminio industriali prodotti dalla Item[1]. Questi prodotti permettono di costruire strutture modulari usando una serie di componenti di base quali profilati di varie dimensioni, squadre, giunti, ed una grande quantità di accessori. Un primo

[1] http://www.item.info

Figura 4.1: TiltOne: vista d'insieme

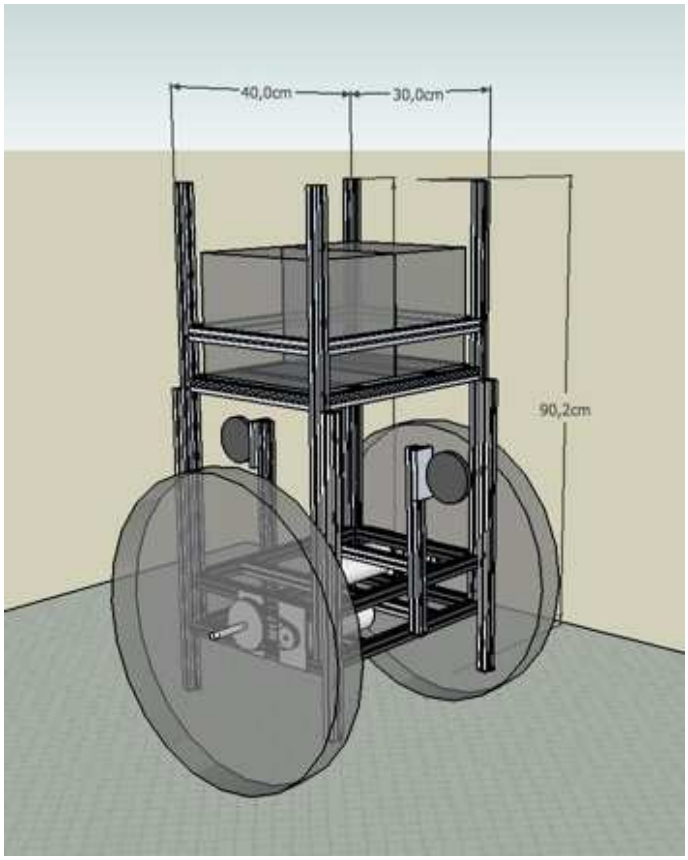


Figura 4.2: Progetto di massima realizzato con Google Sketchup 6

progetto di massima è stato realizzato con Google SketchUp[2], permettendo di definire gli ingombri del robot e la disposizione dei vari elementi. Questo modello è stato utilizzato per ricavare la descrizione della dinamica del sistema (si veda la Sezione 3.1.1) sulla cui base sono state svolte le simulazioni in ambiente Matlab/Simulink (Sezione 3.1.3). Il robot è largo 50cm e profondo 40cm, soddisfando la richiesta di un'impronta a terra contenuta per potersi muovere con agilità in ambienti domestici. L'altezza della base è di 90cm circa, consentendo di montare eventuali dispositivi di interfaccia a livello del busto di un uomo facilitando l'interazione con esso. Il peso complessivo della struttura è di circa 30kg, ed è previsto che si possa trasportre fino a 50kg di carico utile aggiuntivo.

Come si nota in Figura 4.2, sono stati previsti due bracci dotati di ruote alle estremità, la cui altezza è regolabile per impedire che il robot cada

[2]http://sketchup.google.com

Figura 4.3: Progetto dettagliato realizzato con Rhinoceros 4.0

durante le fasi di test.

In seguito si è utilizzato il software CAD Rhinoceros 4.0 [3] per produrre il progetto dettagliato, da cui sono state ricavate le misure dei profilati necessari e la disposizione dei relativi giunti. In questo modo si è potuto procedere con l'ordine delle componenti già tagliate e forate facilitando il lavoro di assemblaggio.

[3] http://www.rhino3d.com

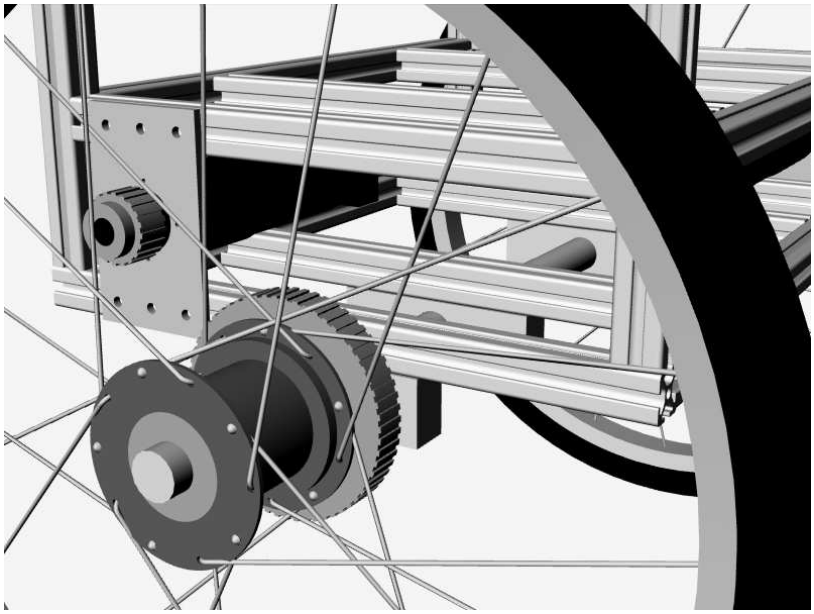


Figura 4.4: Dettaglio della trasmissione

4.2 Ruote e trasmissione

Nel progettare il robot si sono voluti utilizzare componenti il più possibile standard, per ridurre il numero di lavorazioni meccaniche necessarie e per facilitare eventuali modifiche future. In quest'ottica si è scelto di utilizzare per il movimento delle normali ruote a raggi per bicicletta, con cerchi del diametro di 14" e diametro complessivo con pneumatici montati di circa 40 cm.
Le ruote di bicicletta prevedono però una trasmissione a catena dotata di mozzo a scatto libero, non adatta al controllo richiesto da un robot bilanciante a causa dell'elevato gioco. Si è quindi optato per una trasmissione sincrona a ruote dentate e cinghia, per la quale è stato necessario realizzare dei mozzi appositi su cui montare le pulegge. I cerchi sono quindi stati smontati per fissare i raggi al nuovo mozzo. L'insieme della trasmissione è visibile nel disegno riportato in Figura 4.4.

Il disegno del mozzo è stato realizzato con il software CAD Rhinoceros 4.0, tramite cui sono state prodotte le tavole necessarie alla tornitura dei pezzi. I mozzi prevedono le cave per l'alloggiamento di due cuscinetti a sfere inseriti sull'asse delle ruote, realizzato in alluminio. Nelle figure 4.5 e 4.6 sono mostrati rispettivamente una visione prospettica del disegno realizzato

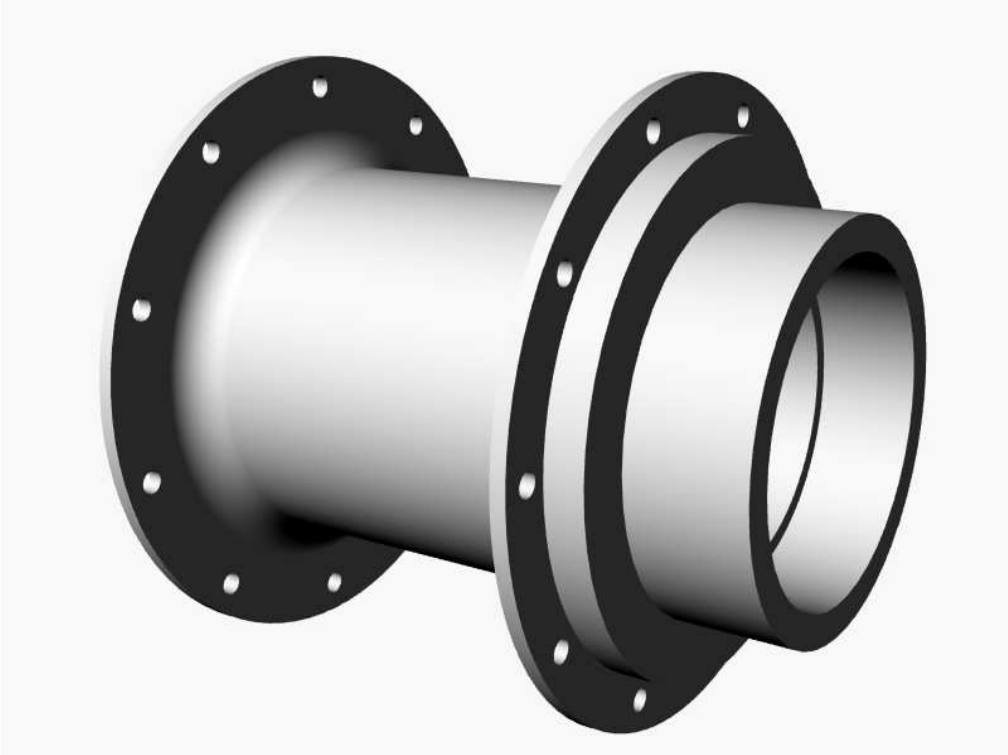


Figura 4.5: Vista prospettica del mozzo

al CAD e una fotografia della ruota montata sul mozzo progettato.

La trasmissione è costituita da cinghie dentate e pulegge Synchroflex[4] della larghezza di 15 mm, capaci di trasferire potenze fino a 5 Kw. Le dimensioni del pignone montato sul motore, della puleggia montata sul mozzo e la lunghezza della cinghia sono state definite in base alle relazioni fornite dal costruttore, che impongono il numero minimo di denti a contatto con la cinghia per garantire il movimento corretto.
In particolare, il numero di denti della cinghia a contatto con il pignone z_e è dato dalla seguente relazione:

$$z_e = \frac{z_1}{180} arccos \frac{(z_2 - z_1)t}{2\pi d} \tag{4.1}$$

in cui z_1 e z_2 indicano rispettivamente il numero di denti della puleggia più piccola (il pignone) e della più grande, t è il passo della cinghia (la distanza tra due denti successivi) e d è la distanza tra i centri delle due ruote dentate.

[4] www.transdev.co.uk

Figura 4.6: Cerchio, mozzo e puleggia assemblati

I motori sono montati su dei supporti in alluminio realizzati appositamente, in grado di scorrere orizzontalmente permettendo di portare in tensione la cinghia senza dover ricorre ad un dispositivo tendicinghia.

4.3 Attuatori

Il robot è mosso da due motori elettrici in corrente continua prodotti da Maxon Motor. Per la scelta delle caratteristiche dei motori sono stati stabiliti i seguenti vincoli:

- il robot deve poter spostare circa 50kg di dispositivi accessori, per un peso complessivo di circa 80kg

- la velocità di spostamento deve essere paragonabile ad una camminata veloce (circa 1.5m/s)
- il robot deve essere in grado di recuperare l'equilibrio fino ad un'inclinazione massima di 10 gradi

Per dimensionare i motori in maniera da soddisfare queste richieste si è ricorsi ad una simulazione in ambiente Matlab/Simulink del sistema, modellizzato come un pendolo inverso infulcrato su un carrello mobile (il modello è mostrato nella Sezione 3.1.3). Si è supposto di utilizzare un controllore PID tarato in modo da poter raggiungere la condizione d'equilibrio partendo dalla condizione pessima di un'inclinazione di 10 gradi rispetto alla verticale. Si è quindi scelto un compromesso tra velocità del raggiungimento del setpoint e contenimento della coppia richiesta, tenendo in considerazione la velocità di spostamento richiesta. Da quest'ultima condizione si è potuta scegliere la riduzione da introdurre tra albero motore e mozzo delle ruote, che producono un avanzamento pari a $2\pi r = 1.3m$ per ogni rivoluzione. I motori Maxon Motor alimentati a 24V hanno velocità di rotazione attorno ai 7500rpm, da cui si ricava che la riduzione necessaria è circa $(7500/60) * (1.3m/s/1.5m/s) = 108$. La riduzione apportata dalle ruote dentate è pari a 4, quindi tra le riduzioni disponibili a catalogo si è scelta quella a 2 stadi con rapporto 26:1.

Inseriti i parametri relativi alla trasmissione nel modello MATLAB/Simulink si è proceduto alla simulazione del sistema ad anello chiuso, i cui risultati sono mostrati in Figura 4.7.
Osservando il grafico si nota che il picco della coppia totale richiesta è pari a circa 350 mN, requisito che ha portato a scegliere dei motori Maxon Motor da 150W, capaci di esercitare una coppia di 170 mN ciascuno.

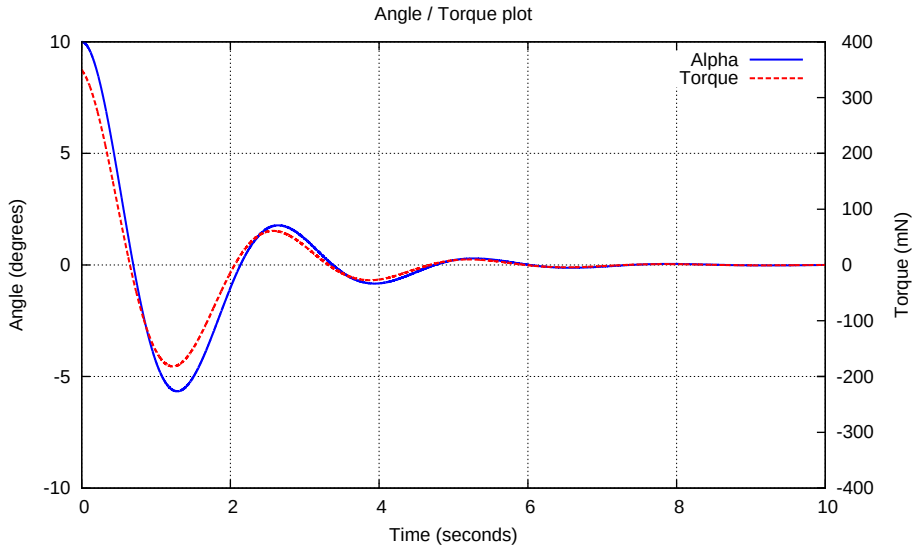


Figura 4.7: Simulazione dell'azione di controllo PID per la stima della coppia richiesta ai motori

Capitolo 5

Progetto elettronico

"Il tempo è ciò che accade quando non accade nient'altro."

La Fisica di Feynman - Richard Feynman

La scheda elettronica sviluppata è costruita attorno ad un microcontrollore, a cui sono affidate tutte le funzioni necessarie alla gestione del robot. Uno schema in cui sono mostrati i vari dispositivi a bordo del robot e la loro connessione alle diverse periferiche del microcontrollore è riportato in Figura 5.1.
Sono presenti dei sensori utilizzati per la stima dell'angolo, una sezione di potenza per il controllo dei motori, dei dispositivi logici per interfacciare il microcontrollore alla sezione di potenza e per interdire il funzionamento dei motori in caso di emergenza e due linee seriali, una via cavo e una wireless, per comunicare con un computer esterno.
Nelle seguenti sezioni vengono descritti dettagliatamente i vari dispositivi utilizzati.

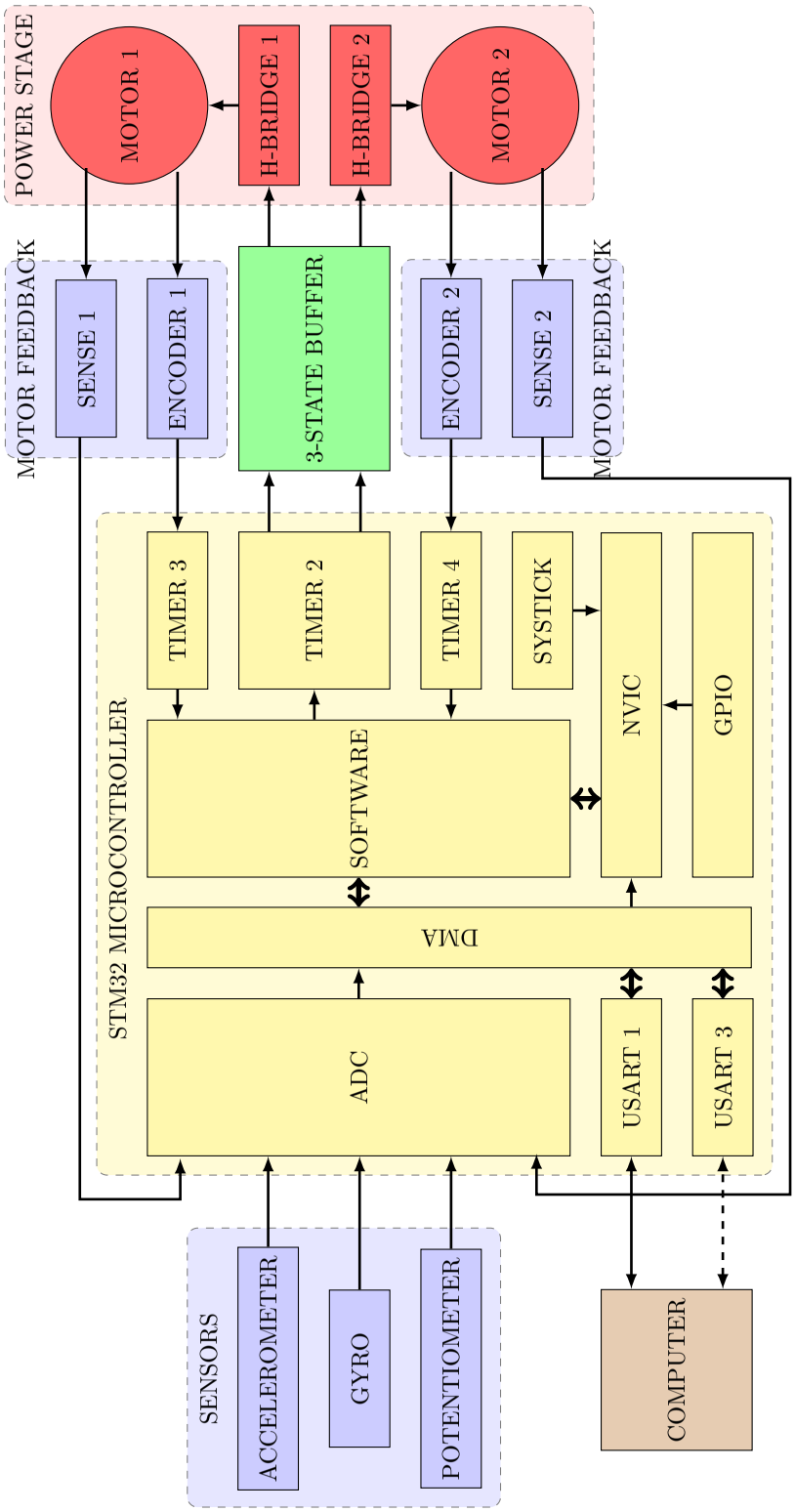


Figura 5.1: Architettura dell'elettronica a bordo del robot

5.1 Sensori

A bordo del robot sono presenti sensori usati per stimare l'angolo rispetto alla verticale e sensori utilizzati per l'odometria.

5.1.1 Accelerometro

Per quanto riguarda la stima dell'angolo vengono utilizzati un accelerometro, che misura l'accelerazione di gravità a cui sono sovrapposte le accelerazioni dovute al moto del robot, ed un giroscopio, la cui misura rappresenta la velocità angolare di rotazione rispetto ad un asse.
Sul mercato sono disponibili numerosi modelli di accelerometri, grazie alle applicazioni in campo *consumer* che negli ultimi anni si sono moltiplicate. Tra gli integrati disponibili la scelta è ricaduta sull'accelerometro a 3 assi LIS344ALH prodotto da ST, che rispetta i requisiti per l'applicazione in oggetto:

- tensione di alimentazione compresa tra 3.3V e 5V
- uscita analogica per poter amplificare e filtrare il segnale prima dell'acquisizione
- fondoscala ridotto a vantaggio di una maggiore risoluzione
- banda maggiore di 100Hz

In tabella 5.1 sono riportate le principali caratteristiche del sensore.

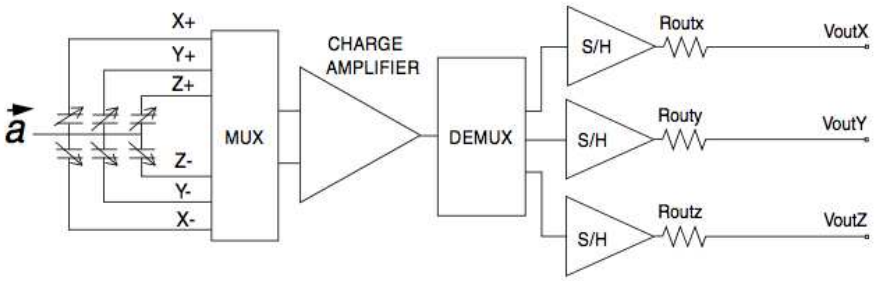


Figura 5.2: Diagramma di funzionamento dell'accelerometro LIS344ALH

L'elemento sensibile dell'accelerometro è un dispositivo in cui tramite tecnologia MEMS delle strutture di silicio sono sospese nel substrato, a cui

Tabella 5.1: Principali caratteristiche dell'accelerometro LIS344ALH

Power Supply	2.4 - 3.6	V
Dynamic Range	±2	g
Sensitivity	0.495	V/g
Nonlinearity	0.5	% FS
Rate Noise Density	50	$\mu g/\sqrt{Hz}$
3dB Bandwidth	up to 500	Hz

sono collegate in maniera da permetterne il movimento lungo la componente dell'accelerazione che si vuole misurare. Lo spostamento della massa sospesa si ripercuote sul valore di due capacità, che in assenza di accelerazione è pari a qualche pF. Lo sbilanciamento tra i due valori, nell'ordine dei fF, viene amplificato tramite un amplificatore di carica, condiviso dai 3 assi, e quindi prelevato da un circuito di *sample* & *hold* che mantiene l'uscita costante fino alla successiva misura. Lo schema di funzionamento è mostrato in Figura 5.2.
Il segnale in uscita è infine filtrato tramite un passa-basso a polo dominante, la cui frequenza di taglio è selezionabile, per ogni asse, scegliendo il valore della capacità C_{load}:

$$f_t = \frac{1}{2\pi R_{out} C_{load}}$$

5.1.2 Giroscopio

La velocità angolare del telaio robot è misurata tramite un giroscopio a singolo asse, posizionato parallelamente all'asse delle ruote. Il modello di giroscopio scelto tra quelli disponibili sul mercato è l'Analog Devices ADXRS150, che soddisfa i requisiti richiesti, simili a quelli elencati per la scelta dell'accelerometro. Le caratteristiche del sensore sono sinteticamente riassunte in tabella 5.2.

Il giroscopio è un dispositivo MEMS in cui due strutture in polisilicio contengono un elemento oscillante portato elettrostaticamente alla risonanza. Questi elementi in movimento si spostano, a causa forza di Coriolis, se sottoposti a rotazione. I due elementi sono collegati a delle dita che costituiscono una delle due armature di un condensatore, la cui capacità varia a

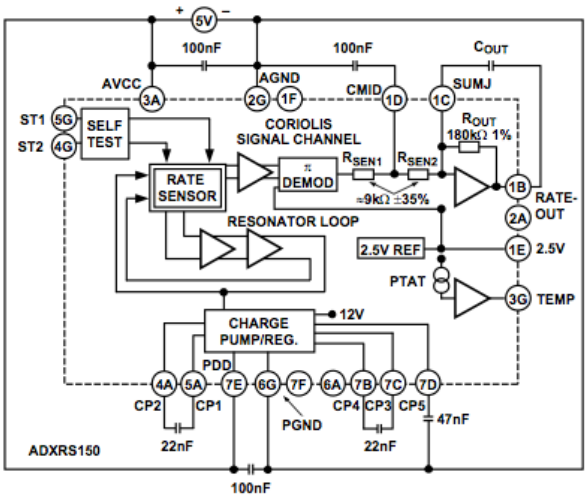


Figura 5.3: Diagramma di funzionamento del giroscopio ADXRS150

Tabella 5.2: Principali caratteristiche del giroscopio ADXRS150

Power Supply	5	V
Dynamic Range	150	$°/s$
Sensitivity	12.5	$mV/°/s$
Nonlinearity	0.1	$\%/V$
Rate Noise Density	0.05	$°/s/\sqrt{Hz}$
3dB Bandwidth	up to 400	Hz

seconda dello spostamento provocato dalle rotazioni. La velocità angolare è quindi misurata tramite un amplificatore di carica, come nel caso dell'accelerometro. Per portare alla risonanza gli elementi sensibili è presente all'interno dell'integrato anche una pompa di carica che genera la tensione necessaria al funzionamento del risonatore elettrostatico. In Figura 5.3 è mostrata la struttura interna del sensore e i componenti esterni necessari al funzionamento.
Il segnale in uscita viene filtrato tramite un passa-basso a polo dominante, la cui frequenza di taglio è selezionabile tramite il condensatore esterno C_{load}:

$$f_t = \frac{1}{2\pi R_{out} C_{load}}$$

5.1.3 Sensori odometrici

A bordo del robot sono infine presenti i sensori di rotazione utilizzati per il controllo dei motori e per l'odometria. Sull'albero di ciascun motore è montato un encoder in quadratura HEDS 5540, prodotto da Agilent. Si tratta di sensori ottici in cui un led illumina un disco, solidale all'albero del motore, che presenta due corone forate disposte come visibile in Figura 5.4. La luce che attraversa il disco viene rilevata da due fototransistor, i cui segnali d'uscita sono amplificati, squadrati e messi a confronto. In Figura 5.5 si nota come i due segnali siano sfasati di $\Pi/2$, permettendo di identificare il verso di rotazione.
Gli encoder HEDS 5540 generano 500 impulsi al giro per ogni canale, permettendo di discriminare 2000 fronti per ogni rotazione dell'albero del motore; il conteggio di questi fronti è eseguito dall'hardware a bordo del microcontrollore, come illustrato nella Sezione 6.2

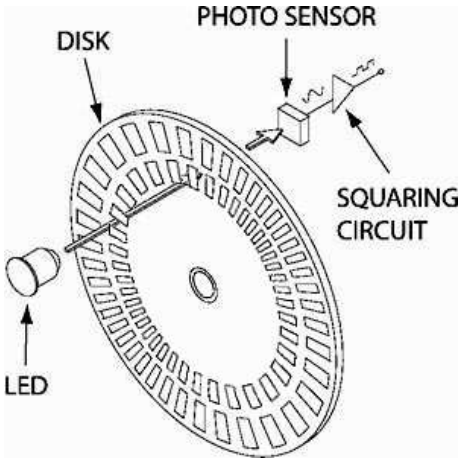
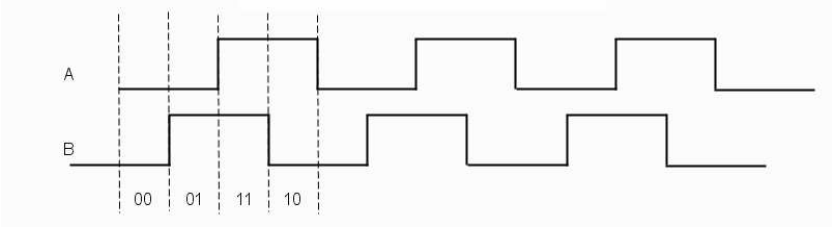


Figura 5.4: Struttura interna di un encoder in quadratura

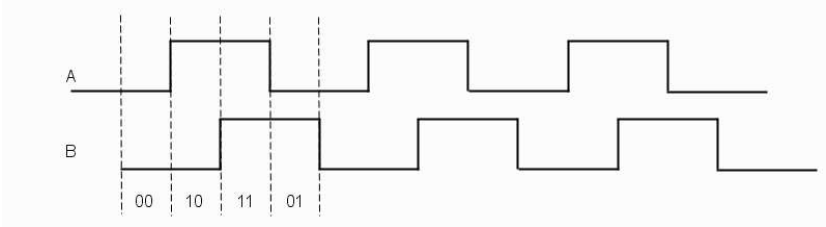
5.2 Condizionamento dei segnali analogici

5.2.1 Cella di Sallen-Key

Si è scelto di utilizzare un accelerometro ed un giroscopio con uscita analogica per poter filtrare il segnale ed amplificarlo nell'intorno della zona di interesse prima dell'acquisizione da parte del microcontrollore. I segnali vengono trattati da un filtro passa basso attivo del secondo ordine, realizzato



(a)



(b)

Figura 5.5: Segnali in uscita da un encoder in quadratura in caso di rotazione oraria (a) ed antioraria (b)

tramite cella di Sallen-Key, la cui struttura generica nella versione a guadagno unitario è mostrata in figura 5.6.

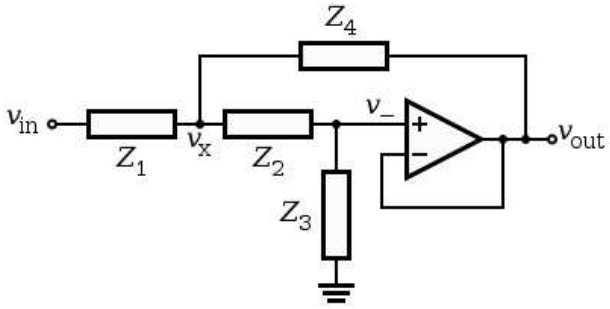


Figura 5.6: Struttura della cella di Sallen-Key

La struttura della cella di Sallen-Key è stata proposta da R. P. Sallen e E. L. Key, del MIT Lincoln Laboratory, nel 1955 [16]. La semplice struttura e la possibilità di ottenere filtri con fattore di qualità Q alto e ampio guadagno in banda hanno reso questa struttura molto diffusa nel campo del condizionamento dei segnali analogici.
Si descrive in seguito il principio di funzionamento del filtro.
Essendo l'operazionale retroazionato negativamente, gli ingressi v_+ e v_-

devono trovarsi allo stesso potenziale. La retroazione a buffer impone quindi:

$$v_+ = v_- = v_{\text{out}} \tag{5.1}$$

Applicando la legge di Kirchhoff per le correnti si può scrivere

$$\frac{v_{\text{in}} - v_x}{Z_1} = \frac{v_x - v_{\text{out}}}{Z_4} + \frac{v_x - v_-}{Z_2} \tag{5.2}$$

e, combinando la 5.1 e la 5.2:

$$\frac{v_{\text{in}} - v_x}{Z_1} = \frac{v_x - v_{\text{out}}}{Z_4} + \frac{v_x - v_{\text{out}}}{Z_2} \tag{5.3}$$

Applicando nuovamente la legge di Kirchhoff per le correnti all'ingresso non invertente dell'amplificatore operazionale si ha:

$$\frac{v_x - v_{\text{out}}}{Z_2} = \frac{v_{\text{out}}}{Z_3} \tag{5.4}$$

che può essere riscritta come:

$$v_x = v_{\text{out}} \left(\frac{Z_2}{Z_3} + 1 \right) \tag{5.5}$$

e, combinando la 5.2 e la 5.5, si trova:

$$\frac{v_{\text{in}} - v_{\text{out}} \left(\frac{Z_2}{Z_3} + 1 \right)}{Z_1} = \frac{v_{\text{out}} \left(\frac{Z_2}{Z_3} + 1 \right) - v_{\text{out}}}{Z_4} + \frac{v_{\text{out}} \left(\frac{Z_2}{Z_3} + 1 \right) - v_{\text{out}}}{Z_2} \tag{5.6}$$

Riscrivendo la 5.6 si ricava la funzione di trasferimento della cella di Sallen-Key:

$$\frac{v_{\text{out}}}{v_{\text{in}}} = \frac{Z_3 Z_4}{Z_1 Z_2 + Z_4(Z_1 + Z_2) + Z_3 Z_4} \tag{5.7}$$

Tramite questa struttura si può realizzare un filtro passa-basso, passa-banda oppure passa-alto scegliendo opportunamente i componenti rappresentati dalle generiche impedenze Z_1, Z_1, Z_2, Z_3.

5.2.2 Implementazione

Nel caso del filtraggio dei segnali di uscita dell'accelerometro e del giroscopio, è necessario realizzare un passa-basso per eliminare il rumore fuori dalla banda di interesse e quindi amplificare il segnale nell'intorno del punto di lavoro al fine di utilizzare tutta la dinamica del convertitore analogico/digitale con cui viene effettuata l'acquisizione.

Per realizzare un filtro passa-basso la struttura generica della cella di Sallen-Key viene modificata operando le seguenti sostituzioni:

$$Z_1 = R_1, \quad Z_2 = R_2, \quad Z_3 = \frac{1}{sC_2}, \quad Z_4 = \frac{1}{sC_1} \tag{5.8}$$

Inoltre la retroazione negativa dell'amplificatore operazionale viene modificata inserendo i resistori R_3 e R_4 che impongono un guadagno in banda pari a:

$$G = 1 + \frac{R_3}{R_4} \tag{5.9}$$

La funzione di trasferimento del filtro è:

$$H(s) = \frac{\overbrace{(1 + \frac{R_3}{R_4})^2}^{G}\overbrace{(2\pi f_c)^2}^{\omega_c^2}}{s^2 + \underbrace{2\pi\frac{f_c}{Q}}_{\frac{\omega_c}{Q}=2\zeta\omega_c} s + \underbrace{(2\pi f_c)^2}_{\omega_c^2}} \tag{5.10}$$

La frequenza di taglio f_c vale quindi:

$$f_c = \frac{1}{2\pi\sqrt{R_1 R_2 C_1 C_2}} \tag{5.11}$$

Il filtro descritto è stato implementato per amplificare segnali in uscita dall'accelerometro e dal giroscopio e per ridurne il rumore. Un circuito analogo viene utilizzato per amplificare e filtrare il segnale prelevato dalle resistenze di sense utilizzate per la misura della corrente che scorre nei motori del robot.

5.3 Elettronica di potenza

5.3.1 Controllo di motori elettrici

A bordo del robot è presente una scheda di potenza utilizzata per il controllo dei motori in corrente continua. Dal punto di vista elettrico, un motore può essere schematizzato come mostrato in Figura 5.7, in cui L indica l'induttanza dovuta agli avvolgimenti, R rappresenta le perdite elettriche del motore , il generatore di tensione E corrisponde alla tensione prodotta dalla rotazione del motore (forza elettromotrice indotta o *BEMF* dall'inglese

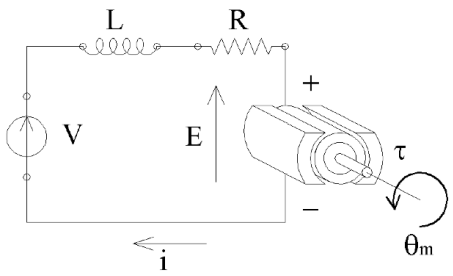


Figura 5.7: Circuito equivalente di un motore a corrente continua

Back Electromotive Force) e V è la tensione applicata. Il motore è descritto dalle seguenti equazioni:

$$E(t) = K_e\dot{\theta}_m(t) = K_e\omega_m(t) \tag{5.12}$$

$$V(t) = E(t) + Ri(t) + L\frac{dI(t)}{dt} \tag{5.13}$$

dove K_e rappresenta la costante elettrica del motore.
Per quanto riguarda la coppia esercitata dal motore vale la relazione:

$$\tau = K_t i(t) \tag{5.14}$$

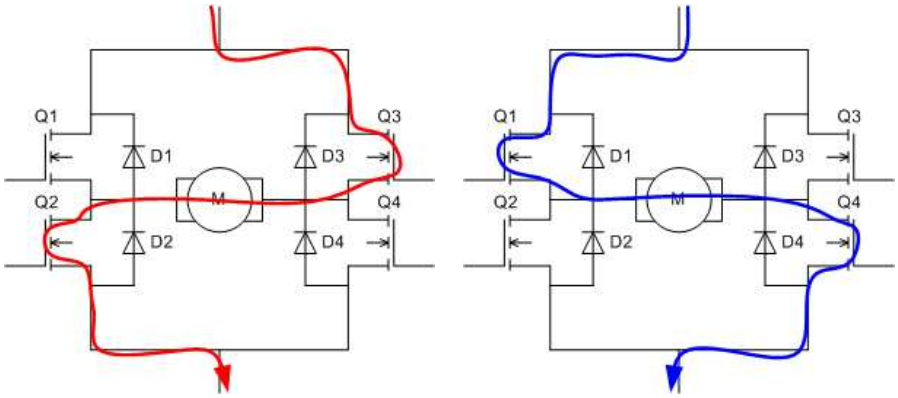
in cui K_t è la costante di coppia del motore.
Osservando le relazioni precedenti si nota il legame tra BEMF E e velocità di rotazione e tra corrente assorbita i e coppia esercitata τ. Si possono immaginare due casi estremi:

- se il motore ruota senza essere collegato ad un carico, la coppia richiesta τ è idealmente nulla e la corrente i è pari a zero; la tensione ai capi del motore è quindi pari ad E
- se il motore è in stallo, ovvero il rotore è bloccato, la BEMF generata è nulla e la tensione ai capi del motore è dovuta solo alla corrente i di stallo e alle perdite elettriche R (idealmente, $R = 0$ quindi la tensione ai capi del motore è anch'essa nulla)

La tecnica più diffusa per il controllo dei motori in corrente continua è la modulazione PWM (dall'inglese *Pulse Width Modulation*), che consiste nell'applicare ai capi del motore un'onda quadra di tensione pari a V_H per un certo periodo t_{ON} e a V_L per il restante periodo $t_{OFF} = T_S - t_{ON}$, avendo chiamato T_S la durata del ciclo. Il rapporto tra t_{ON} e t_S prende il nome di *duty cycle* D:

$$D = \frac{t_{ON}}{t_S} \tag{5.15}$$

Per poter pilotare un motore in entrambe le direzioni di rotazione si ricorre allo schema circuitale denominato *ponte H*, a causa della disposizione delle connessioni, in cui quattro interruttori possono collegare ciascun terminale del motore alla tensione V_H(generalmente pari alla tensione di alimentazione) oppure a V_L (generalmente pari alla massa). I due interruttori che portano il potenziale a V_H sono denominati *high-side*, mentre quelli che portano il potenziale a V_L vengono chiamati *low-side*. In Figura 5.8 è



(a) (b)

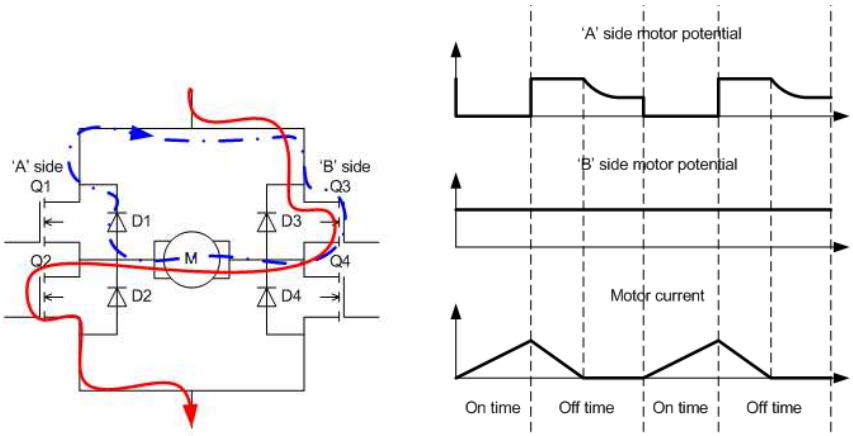
Figura 5.8: Schema di un generico ponte H e flusso della corrente per ottenere i due versi di rotazione

mostrato lo schema di un generico ponte H e il percorso della corrente nel caso di rotazione in un verso o nell'opposto: se Q2 e Q3 sono in conduzione, il morsetto di sinistra del motore è connesso a massa, mentre il morsetto di destra è connesso alla tensione di alimentazione; viceversa, se ad essere accesi sono Q1 e Q4, il motore vede ai suoi capi una tensione opposta alla precedente, ruotando quindi nell'altra direzione. Modulando i segnali di comando degli interruttori tramite la tecnica PWM si possono ottenere velocità $\dot{\theta}_m$ (o valori di coppia τ) variabili.

I ponti H possono essere pilotati secondo schemi differenti: mentre durante la fase t_{ON}, in generale, prevede sempre che sia acceso un interruttore nella parte high-side e l'interruttore low-side opposto, la fase t_{OFF} può essere gestita vari modi. Dato che le configurazioni di cross-conduzione, ovvero quando conducono contemporaneamente Q1 e Q2 oppure Q3 e Q4, sono ovviamente da evitare, le configurazioni restanti per ciascun ramo (destro e sinistro) sono quelle per cui conduce l'high-side, oppure conduce il low-side, oppure sono entrambi spenti. La gestione della fase t_{OFF} differenzia le differenti modalità di funzionamento.

In seguito sono mostrate le due differenti tipologie utilizzate per il controllo dei motori del robot.

La tecnica di controllo di un ponte H più diffusa, ed anche più semplice da implementare, è denominata modalità *Sign/Magnitude*. Il flusso della corrente nel circuito durante le fasi t_{ON} e t_{OFF} e il grafico delle tensioni ai capi del motore e della corrente che scorre negli avvolgimenti sono mostrati in Figura 5.9. Durante il periodo t_{ON} un interruttore high-side e il suo op-



(a) (b)

Figura 5.9: Modalità PWM Sign/Magnitude: (a) flusso della corrente durante il periodo t_{ON} (linea continua) e durante t_{OFF} (linea tratteggiata) (b) grafico delle tensioni ai capi del motore e della corrente assorbita

posto low-side sono accesi, ad esempio Q3 e Q2. La corrente fluisce quindi dal ramo B (a destra) del ponte verso il ramo A, imponendo un verso di rotazione al rotore. Durante il periodo t_{OFF} l'interruttore Q3 rimarrà acceso, mentre Q2 viene interdetto. In questo caso la corrente, che non può annullarsi istantaneamente a causa del comportamento induttivo del motore, fluirà attraverso il diodo di ricircolo D1, polarizzanto in diretta (la corrente di polarizzazione diretta di D2 è invece orientata nel verso opposto). La tensione ai capi del motore V_M, in questa fase, vale quindi:

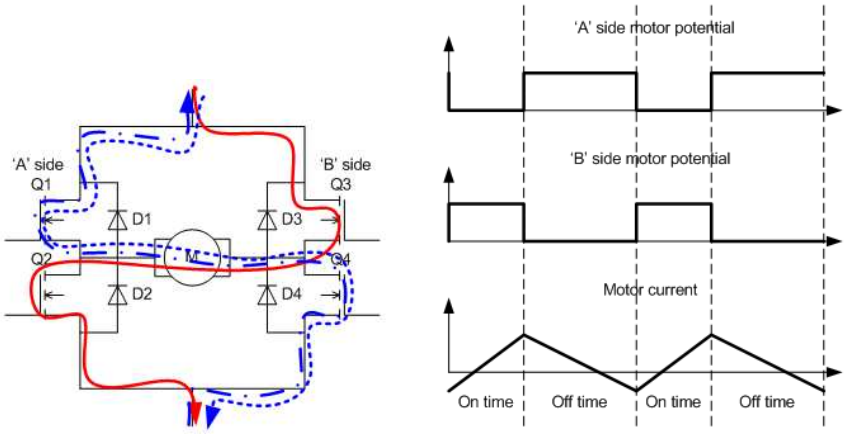
$$V_M = V_F \tag{5.16}$$

avendo indicato con V_F la tensione di polarizzazione diretta del diodo D1. La tensione V_C ai capi dell'avvolgimento vale, trascurando le perdite elettriche rappresentate da R e in assenza di carico:

$$V_C = E + V_F - iR \simeq E \tag{5.17}$$

In questa modalità la corrente i che scorre nel motore, la cui variazione è legata alla tensione ai capi dell'induttanza secondo la legge $V_L = L\frac{di}{dt}$, decresce lentamente (*slow decay mode*), e il diodo D1 rimane polarizzato a lungo. Quando la corrente si annulla, D1 si interdice e la tensione V_M ai capi del è pari a V_G fino all'inizio del ciclo successivo.
In questa modalità il diodo di ricircolo conduce per tutto il periodo in cui la corrente non è nulla, e la potenza da esso dissipata può raggiungere valori significativi. È quindi necessario utilizzare diodi con V_F bassa per limitare la dissipazione e non rischiare di danneggiare il dispositivo. Un altro problema di questa modalità è che durante la fase di conduzione del diodo di ricircolo la corrente scorre nel motore, in Q3 e in D1 formando un anello interno al ponte H, rendendo difficile quantificarne il valore.

Una tecnica di controllo alternativo, chiamata *Locked Antiphase*, prevede invece che ci siano sempre due interruttori accesi attraverso cui la corrente può scorrere. Il flusso della corrente nel circuito durante le fasi t_{ON} e t_{OFF} e il grafico delle tensioni ai capi del motore e della corrente che scorre negli avvolgimenti sono mostrati in Figura 5.10. Durante il periodo t_{ON} gli in-



(a) (b)

Figura 5.10: Modalità PWM Locked Antiphase: (a) flusso della corrente durante il periodo t_{ON} (linea continua) e durante t_{OFF} (linea tratteggiata) (b) grafico delle tensioni ai capi del motore e della corrente assorbita

terruttori Q2 e Q3 sono accesi, mentre durante il periodo t_{OFF} sono i loro complemetari Q1 e Q4 ad essere in conduzione.
La tensione V_M presente sui terminali del motore durante il periodo t_{OFF} è ora pari alla tensione di alimentazione:

$$V_M = V_{ALIM} \tag{5.18}$$

La tensione ai capi dell'avvolgimento vale quindi:

$$V_C = E + V_{ALIM} - iR \simeq E + V_{ALIM} \tag{5.19}$$

In questa modalità, essendo la tensione ai capi dell'induttanza del motore molto maggiore, la corrente decresce più rapidamente (*fast decay mode*) e i diodi di ricircolo non sono praticamente mai in conduzione, eccezion fatta per il breve periodo che separa lo spegnimento degli interruttori attivi durante il periodo t_{ON} e l'accensione dei loro complementari, riducendo notevolmente la quantità di energia che essi devono dissipare.
Il percorso della corrente, inoltre, passa sempre da massa e dalla tensione di alimentazione, permettendo di misurarne l'entità mediante una resistenza di *sense*.

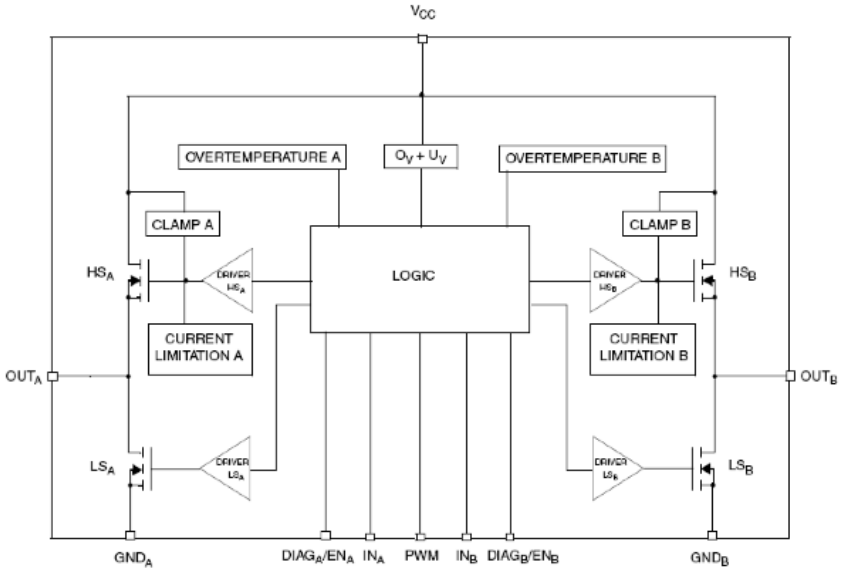
Per il controllo dei motori dei robot sono stati realizzati due circuiti di potenza che implementano entrambe le tecniche di pilotaggio descritte.
Il primo circuito realizzato è basato su una soluzione completamente integrata, particolarmente resistente ai fenomeni di sovracorrente e surriscaldamento e caratterizzata da un basso costo di realizzazione.
La seconda souzione è invece basata su un driver per ponti H utilizzato per il pilotaggio di quattro MOSFET di potenza discreti, permettendo maggior libertà nella scelta dei componenti e della tecnica di modulazione PWM, tramite cui è possibile implementare anche un controllo in coppia (ovvero corrente) dei motori.
Come mostrato nel Capitolo 7 i due ponti H realizzati vengono utilizzatti abbinati a differenti controllori per il bilanciamento del robot.

5.3.2 Driver PWM in modalità Sign/Magnitude

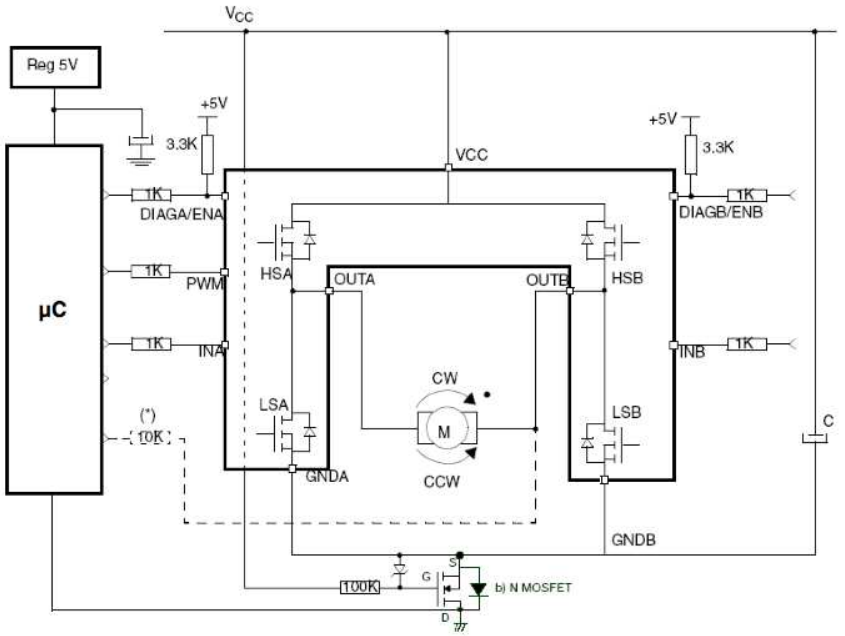
Al fine di realizzare una scheda di potenza a basso costo per il controllo motori si è scelto di utilizzare un driver completamente integrato prodotto da ST, che racchiude proprio interno sia la logica di controllo che i MOSFET di potenza. Il circuito integrato in oggetto è il VNH3SP30, la cui architettura interna è mostrata in Figura 5.11(a).

In un singolo package il driver VNH3SP30 racchiude:

- la logica di controllo, che gestisce l'accensione e lo spegnimento dei MOSFET high-side e low-side a seconda del valore dei segnali di ingresso



(a)



(b)

Figura 5.11: Ponte H integrato VNH3SP30: (a) architettura interna (b) applicazione tipica

- la protezione *overvoltage* e *undervoltage* per spegnere il dispositivo in caso la tensione di alimentazione sia fuori dalla regione di funzionamento ($5.5 - 36$ V)
- i circuiti di *clamp* per la protezione dei MOSFET dalle alte tensioni tra drain e source
- i driver per il pilotaggio di ciascun transistor di potenza
- il circuito limitatore di corrente, che modula la tensione di gate dei transistor high-side in caso venga raggiunta la soglia
- la protezione dal surriscaldamento, che agisce anch'essa sulle tensioni di gate dei transistor high-side
- il circuito di *fault detection* che comunica l'occorrere di configurazioni anormali mediante lo stato dei piedini EN/DIAG

Le principali caratteristiche del dispositivo sono mostrate in Tabella 5.3.

Tabella 5.3: Principali caratteristiche del ponte H VNH3SP30

V_{CCmax}	40	V
I_{OUTmax} (continuous)	30	A
I_{OUTmax} (pulsed)	30	A
R_{DSON},low-side	23	mΩ
R_{DSON},high-side	11	mΩ
V_F $(I_D = 12A)$	0.8	V
f_{max}	10	Khz

Il vantaggio principale nell'utilizzare una soluzione integrata di questo tipo risiede nella semplicità di realizzazione (sono necessari pochi componenti esterni, come mostrato in Figura 5.11(b)) e nella dotazione di dispositivi di protezione che garantiscono il corretto funzionamento del circuito.
Una soluzione completamente integrata limita però le modalità di controllo PWM utilizzabili alle sole compatibili con caratteristiche del dispositivo, determinate dalle scelte del costruttore. I ponti H ST di questa famiglia impongono ad esempio, tramite la circuiteria interna, un tempo morto t_{DEL} in cui tutti i MOSFET sono interdetti ogni qualvolta si voglia invertire il senso di rotazione del motore. Questo periodo, della durata di circa $0.6 - 1.8\mu S$, fa si che sia impossibile utilizzare il ponte H in modalità locked antiphase, in cui il verso di rotazione è continuamente invertito (si veda la sezione precedente). La scheda di controllo motori basata su questa soluzione prevede

quindi di essere pilotata in modalità sign/magnitude senza feedback di corrente, ed è utilizzata per lo sviluppo del controllore basato su apprendimento per rinforzo descritto nella Sezione 7.3.

5.3.3 Driver PWM in modalità Locked Antiphase

Parallelamente alla soluzione descritta nella sezione precedente è stata sviluppata una scheda per il controllo dei motori basata su un driver per ponti H prodotto da Allegro Micro, modello A3941, e quattro MOSFET di potenza IRF7862.
L'integrato A3941 permette di pilotare 4 MOSFET a canale N per realizzare un ponte H completo, racchiudendo in un unico package gran parte degli elementi necessari ma lasciando al progettista la libertà di scegliere i dispositivi di potenza a seconda dell'applicazione.
L'architettura interna dell'integrato è riportata in Figura 5.12.
Il driver implementa i seguenti dispositivi:

- la logica di controllo per la gestione dell'accensione dei segnali di gate, l'impostazione dei tempi di commutazione nelle varie transizioni e la selezione della modalità di funzionamento
- un regolatore di tensione stabilizzato per l'alimentazione della logica interna e di eventuali dispositivi esterni
- la circuiteria per la protezione dal surriscaldamento del driver e per la diagnosi di eventuali anomalie
- le pompe di carica tramite cui vengono prodotte le tensioni necessarie all'accensione dei gate degli N-MOSFET high-side
- i driver per fornire la corrente necessaria all'accensione e allo spegnimento rapido dei MOSFET per permettere di operare ad alta frequenza

Le caratteristiche principali dell'integrato sono riportate in Tabella 5.4.

I transistor utilizzati sono MOSFET verticali, con diodo di ricircolo integrato, prodotti dalla International Rectifier sfruttando la teconologia proprietaria HEXFET: essa è un'estensione del concetto di struttura interdigitata per la realizzazione del gate, utilizzata per aumentare la lunghezza di canale W in rapporto alla superficie occupata. La tecnologia HEXFET

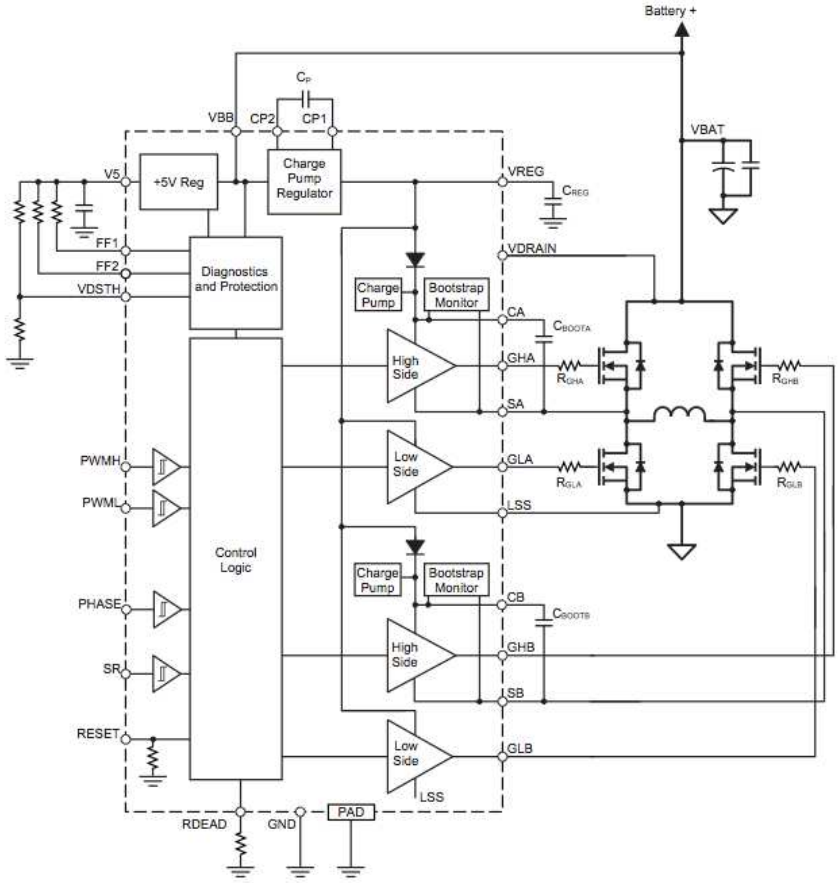


Figura 5.12: Architettura del driver A3941

consiste nel realizzare una struttura composta da celle di forma esagonale, come mostrato in Figura 5.13, massimizzando la lunghezza di canale W, ovvero il perimetro delle aree di gate, consentendo di trasportare una grande quantità di corrente anche in dispositivi di dimensioni ridotte come quello in oggetto.
Caratteristica interessante del dispositivo è anche la R_{DSON} molto ridotta, nell'ordine di qualche mΩ. Un percorso conduttivo così poco resistivo è ottenuto mediante il compromesso con il valore della tensione di breakdown tra source e drain, che comunque non rappresenta un problema per l'applicazione in oggetto.
Le caratteristiche principali del dispositivo sono riportate in Tabella 5.5.

Gli ingressi dell'integrato permettono di impostare la modalità di commutazione del ponte H, a seconda di come vengono configurati:

Tabella 5.4: Principali caratteristiche del driver per ponti H A3941

V_{CCmax}	50	V
Pull-up ON resistance	8	Ω
Pull-down ON resistance	3	Ω
f_{max}	250	Khz

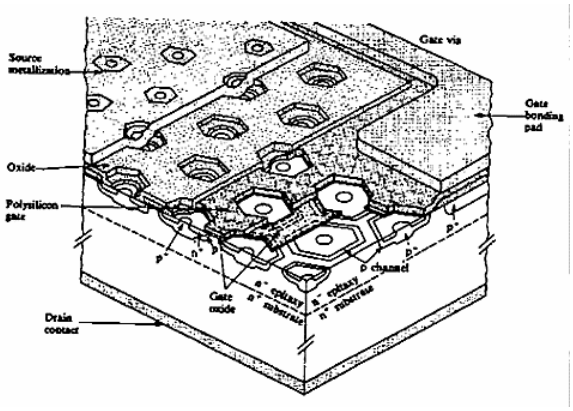


Figura 5.13: Struttura di un dispositivo HEXFET

RDEAD permette di controllare la durata del tempo morto necessario per evitare situazioni di cross-conduzione nel ponte H

PWMH e PWML controllano rispettivamente il transistor attivo della parte high-side e low-side del ponte

PHASE determina il verso in cui scorre la corrente nel carico, ovvero la direzione di rotazione del motore; assieme a PWMH e PWML permette di pilotare il motore in tutti i quattro quadranti

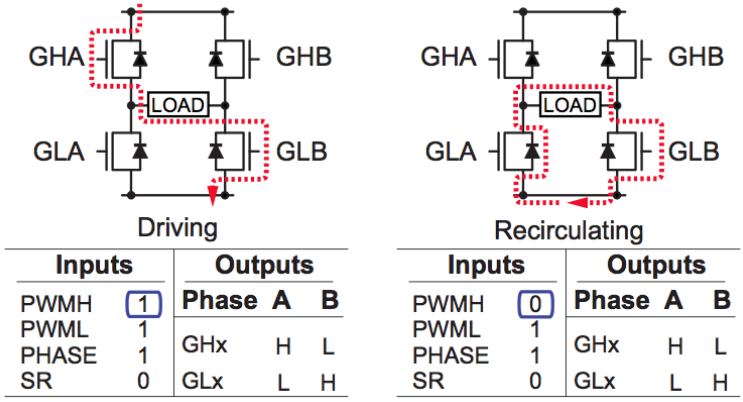
SR abilita e disabilita la rettificazione sincrona: quando essa viene attivata, ogni volta che durante un ciclo viene spento un transistor, il driver provvede ad accendere quello opposto per far si che la corrente generata dal motore scorra in esso invece che nel diodo di ricircolo,

Tabella 5.5: Principali caratteristiche del MOSFET IRF7862

V_{CCmax}	30	V
I_{OUTmax} (continuous)	21	A
I_{OUTmax} (pulsed)	170	A
R_{DSON} ($V_{GS} = 10V$)	3.7	mΩ
Q_g	30	nC

riducendo la dissipazione su quest'ultimo; viceversa quando SR è basso i cicli di accensione e spegnimento dei MOSFET di potenza sono minori, riducendo la dissipazione del driver.

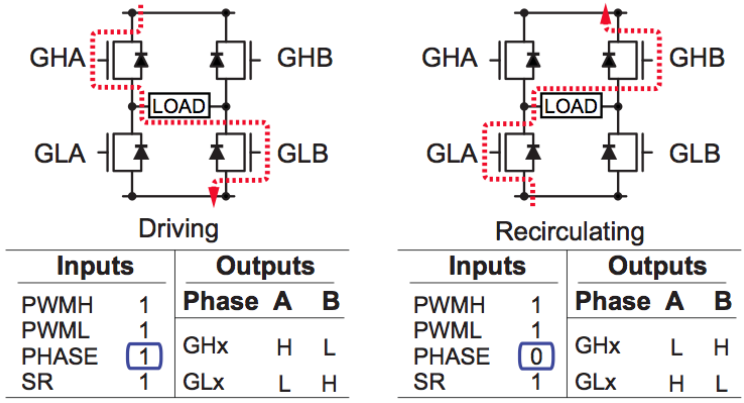
Controllando opportunamente questi ingressi è possibile pilotare il ponte H utilizzando diverse tecniche, tra cui quelle descritte in precedenza ed utilizzate per il controllo dei motori del robot (in Figura 5.14 sono mostrate le configurazioni usate).



Inputs		Outputs		
PWMH	1	Phase	A	B
PWML	1	GHx	H	L
PHASE	1	GLx	L	H
SR	0			

Inputs		Outputs		
PWMH	0	Phase	A	B
PWML	1	GHx	H	L
PHASE	1	GLx	L	H
SR	0			

(a)



Inputs		Outputs		
PWMH	1	Phase	A	B
PWML	1	GHx	H	L
PHASE	1	GLx	L	H
SR	1			

Inputs		Outputs		
PWMH	1	Phase	A	B
PWML	1	GHx	L	H
PHASE	0	GLx	H	L
SR	1			

(b)

Figura 5.14: Driver per ponti H A3941: (a) modalità PWM sign/magnitude (b) modalità PWM locked antiphase

Si può notare come nella modalità locked antiphase la corrente scorra attraverso l'alimentazione e la massa anche durante la fase di ricircolo, po-

tendo quindi essere misurata attraverso una resistenza di sense, che nella scheda realizzata è inserita tra i source dei MOSFET low-side e massa.

Per poter implementare un controllo in corrente accurato è importante che il ripple che essa presenta sia contenuto. La stabilizzazione della corrente riduce anche le vibrazioni del motore, prolungandone la vita.
Il ripple della corrente che scorre nel motore, supponendo che il periodo T_S di un ciclo sia molto minore della costante elettrica data dall'induttanza del motore e dalla sua resistenza serie, è approssimabile dalla seguente relazione:

$$\delta I_{ppM} = \frac{VT_S}{2L} \tag{5.20}$$

Si è quindi imposto di mantenere il ripple entro il 10% della corrente massima inserendo un'induttanza in serie al motore.

5.4 Logica e microcontrollore

Il microcontrollore utilizzato è un STM32 prodotto da ST, basato su architettura ARM 32-bit CORTEX-M3. Le principali caratteristiche del dispositivo sono:

- architettura a 32 bit
- frequenza massima di funzionamento 72 Mhz
- moltiplicazione e divisione a singolo ciclo
- 512Kb di memoria Flash e 64Kb di memoria SRAM
- NVIC (Nested Vector Interrupt Controller)
- alimentazione tra 2.0 e 3.6 V
- 2 ADC a 12bit, fino a 16 canali
- controllore DMA a 12 canali
- 50 porte GPIO (General Purpose Input/Output)
- 11 timer
- 2 interfacce i2c, 3 USART, 3 SPI, CAN bus, USB e SDIO

L'architettura del microcontrollore è mostrata in Figura 5.15.

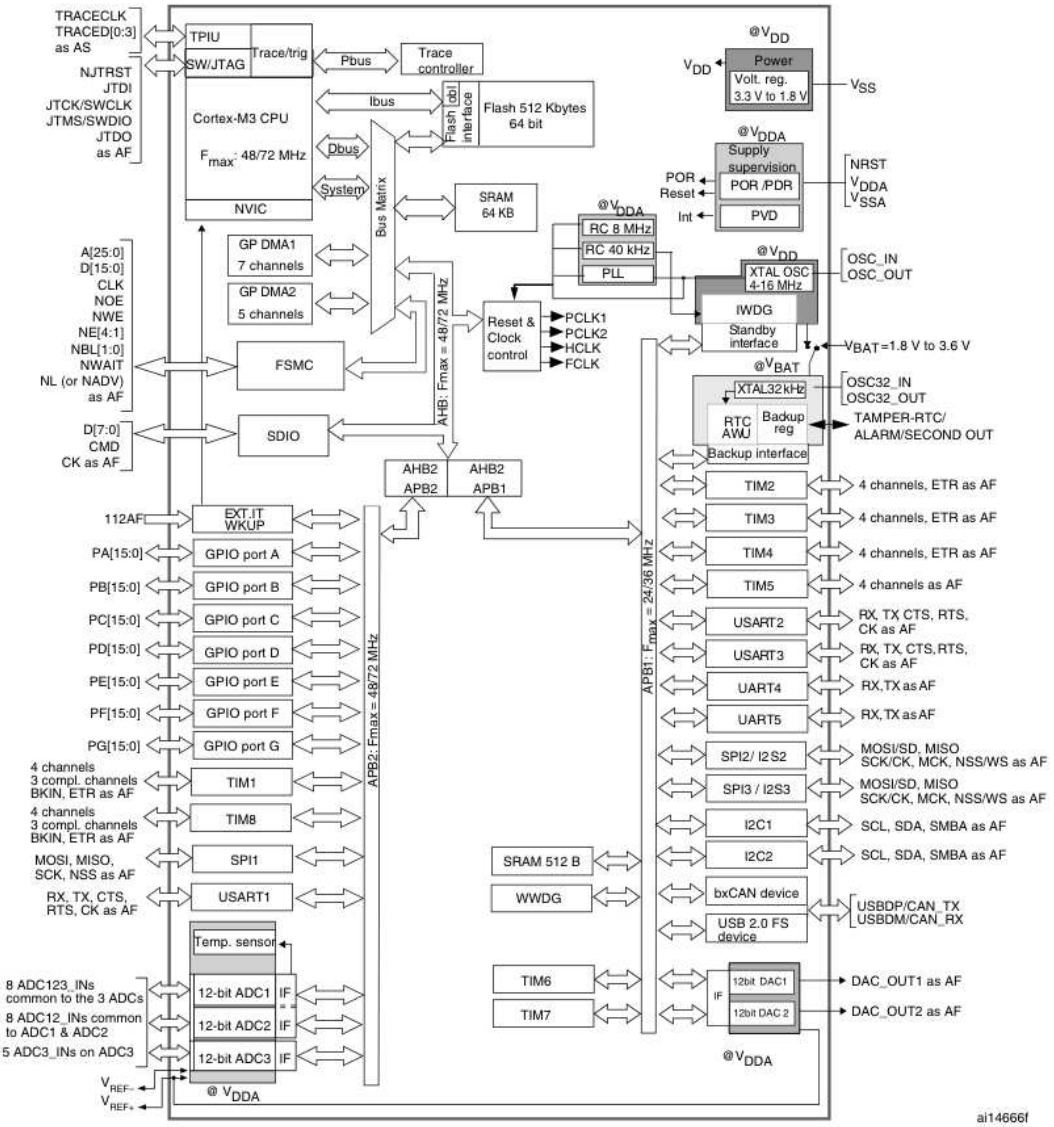


Figura 5.15: Architettura del microcontrollore STM32

Il microcontrollore viene utilizzato alla massima frequenza di 72 Mhz, tramite un quarzo a 8 Mhz a cui viene agganciato il PLL interno che permette di moltiplicare tale frequenza di un fattore 9. Vengono sfruttate le seguenti periferiche:

- 2 canali ADC per l'acquisizione dei segnali generati da accelerometro e giroscopio
- 2 canali ADC per il current sense
- 2 timer per i due encoder ottici montati sui motori
- 1 timer per la generazione dei segnali PWM utilizzati per il controllo dei motori
- 2 interfacce USART per le comunicazioni con l'esterno (via cavo seriale e senza fili tramite modulo ZigBee)
- DMA per il trasferimento dei dati da ADC a memoria SRAM e per l'uso delle periferiche USART

La logica a bordo del robot è è completata da un buffer tri-state interposto tra il microcontrollore e i ponti H.
Il segnale di buffer disable è collegato a due "baffi" di sicurezza che toccano terra nel caso il robot stia cadendo, inibendo il funzionamento dei motori come misura di sicurezza. Durante le normali operazioni invece il buffer è trasparente e funge da level shifter per i segnali PWM e DIR in uscita dal microcontrollore, funzionante a 3.3 V, che vengono utilizzati dai ponti H, funzionanti a 5 V.

5.5 Comunicazioni

Il robot è predisposto per comunicare con l'esterno attraverso porta seriale RS232 e collegamento senza fili Zigbee.

La comunicazione seriale è gestita tramite una delle periferiche USART presenti all'interno del microcontrollore, la cui configurazione sarà presentata nella Sezione 6.4. Il driver della porta seriale e' il noto MAX232 prodotto da Maxim, che integra le pompe di carica necessarie a pilotare la linea RS232, funzionante con livelli di tensione pari a $\pm 12V$, con i segnali logici generati dal microcontrollore, funzionante a $3.3V$.

La comunicazione wireless è affidata ad un modulo Xbee Pro prodotto da Digi configurato per funzionare in modalità punto-punto. Il modulo, funzionante a 3.3V e quindi pilotabile direttamente dal microcontrollore, permette di trasmettere via radio, secondo lo standard IEEE 802.15.4, i segnali generati da un'altra delle periferiche USART, che vengono quindi ricevuti da un modulo USB collegato ad un computer esterno. In questo modo è possibile comunicare col robot e pilotarlo senza il vincolo dei cavi.

Capitolo 6

Software

"Credo che alla fine del secolo l'uso delle parole e l'opinione delle persone di cultura saranno cambiate a tal punto che si potrà parlare di macchine pensanti senza aspettarsi di essere contraddetti."

Alan Mathison Turing

Il software a cui è affidato il controllo del robot è scritto in linguaggio ANSI C e compilato per essere eseguito sull'architettura CORTEX-M3 su cui il microcontrollore ARM STM32 è basato. Il software si occupa dell'acquisizione dei segnali prodotti dai sensori, dell'esecuzione dell'algoritmo di controllo dell'equilibrio e dell'esecuzione dei comandi esterni. Nelle seguenti sezioni è illustrata l'implementazione delle diverse funzioni necessarie alla gestione del robot.

6.1 Architettura

Il software per il controllo del robot è stato scritto sulla base di alcuni concetti fondamentali:

- gli algoritmi di controllo, l'output dei dati ricevuti dai sensori e il comando dei dispostivi connessi al microcontrollore richiedono un'esecuzione periodica, più o meno frequente a seconda del tipo di azione. Queste funzioni quindi devono essere invocate ad intervalli di tempo

precisi, scanditi tramite la generazione di interrupt da parte di un timer

- ovunque sia possibile, si predilige l'utilizzo del controllore DMA per eseguire il maggior numero possibile di operazioni in *background*, riducendo il carico di lavoro del microcontrollore

- il software e gli algoritmi di controllo devono essere configurabili durante l'esecuzione del programma, limitando il più possibile le operazioni di programmazione della memoria programma del microcontrollore

L'esecuzione dei task periodici è gestita tramite un timer, mantenuto costantemente in esecuzione permettendo di generare un interrupt ogni millisecondo. Tra i timer disponibili si è scelto di utilizzare il *SYSTICK timer*, un *down-counter* a 24 bit dedicato ai sistemi operativi real time [17] che genera un interrupt quando il conteggio giunge a zero.
Ad ogni invocazione della routine di interrupt viene incrementato un contatore e si verifica se sia il momento di eseguire qualcuno tra i cicli di controllo impostati. Sono presenti cicli eseguiti a differenti frequenze, ognuno dei quali richiama l'esecuzione di differenti compiti:

1 Hz : questo ciclo è utilizzato esclusivamente per funzioni di debug, come far lampeggiare costantemente un LED per indicare che il programma è in esecuzione, eseguire serie predefinite di comandi per verificare il corretto funzionamento delle periferiche e degli algoritmi di controllo ed inviare su porta seriale l'elenco dei dati che vengono stampati in output, come discusso nella Sezione 6.4

10 Hz : questo ciclo gestisce l'aggiornamento del display LCD

50 Hz : in questo ciclo viene eseguito l'algoritmo per il controllo della stabilità del robot. Sono disponibili diversi algoritmi, attivabili e configurabili *runtime*, per i quali può essere attivata la modalità di debug o l'aggiunta di rumore bianco additivo. Vengono inoltre inviati alla periferica USART i dati ricevuti dai sensori e lo stato del robot

100 Hz : a questa frequenza viene eseguito l'update del filtro di Kalman utilizzato per la stima dell'angolo del robot rispetto alla verticale

1 Khz : il ciclo più veloce viene utilizzato per l'aggiornamento dei dati odometrici mediante la lettura degli encoder ottici e per il controllore

PID utilizzato per inseguire il setpoint di velocità o di corrente dei motori

Il controllore DMA è configurato per trasferire in continuazione il risultato delle conversioni dell'ADC in una struttura di dati apposita, come mostrato nella Sezione 6.2, permettendo a qualsiasi parte del programma di accedere all'output dei sensori semplicemente leggendo il valore contenuto nelle variabili.
Il DMA è utilizzato anche per trasferire il contenuto del buffer software della porta seriale alla periferica hardware, alleggerendo il carico di lavoro del microcontrollore. La gestione delle comunicazioni seriali viene trattata nella Sezione .

I parametri di configurazione del software e lo stato del robot sono contenuti in due apposite strutture, denominate `Config` e `Status`.
La struttura `Config` raggruppa tutti i parametri che può essere necessario modificare durante mentre il software è in esecuzione, ovvero:

- la configurazione generale del robot, ovvero la scelta delll'algoritmo di bilanciamento tra quelli disponibili e l'attivazione o inibizione dei motori
- i valori dell'*offset* dei sensori, riprogrammabili durante l'esecuzione del programma per impostare lo zero dell'angolo e della velocità angolare, e il fattore di scala da applicare al dato letto dall'ADC per interpretare le misure effettuate
- la configurazione relativa ai motori: selezione dei canali PWM utilizzati e del verso di rotazione, associazione dei canali encoder a ciascun motore, impostazione della dead band e selezione del metodo di controllo (si rimanda alla Sezione 6.3)
- le costanti relative ai vari algoritmi di controllo attivi

La struttura `Status` racchiude invece tutti i dati relativi allo stato del robot, che vengono aggiornati durante l'esecuzione del programma, ovvero:

- valori dei segnali acquisiti dall'ADC, aggiornati in continuazione tramite l'utilizzo del controller DMA
- angolo del robot rispetto alla verticale e velocità angolare, costantemente aggiornati dal filtro di Kalman

- informazione relative ai motori (velocità di rotazione, corrente assorbita, stato dei ponti H)
- stato dei controllori attivi (controllo di bilanciamento, controllo di traiettoria, controllori dei motori)
- setpoint impostati (angolo o velocità da inseguire)

Tutte le componenti del software possono sempre accedere a queste due strutture, dichiarate globali, permettendo di conoscere la configurazione del sistema, il suo stato e i dati provenienti dai sensori semplicemente attraverso la lettura di una determinata variabile.

6.2 Acquisizione dei segnali

I segnali acquisiti dal microcontrollore sono suddivisibili in due categorie:

- segnali analogici, filtrati ed amplificati come descritto nella Sezione 5.2, convertiti in valori digitali utilizzando l'ADC interno al microcontrollore
- segnali digitali, acquisiti mediante periferiche apposite o utilizzati per la generazione di un interrupt

6.2.1 Segnali analogici

I segnali di tipo analogico sono generati dai seguenti sensori a bordo del robot:

- l'accelerometro, utilizzato per la stima dell'angolo del telaio del robot rispetto alla direzione dell'accelerazione di gravità
- il giroscopio, tramite cui viene misurata la velocità angolare del telaio attorno all'asse delle ruote
- le resistenze di *sense* utilizzate per misurare la corrente arrorbita da ciascun motore
- un potenziometro utilizzato in fase di debug per ottenere una misura attendibile dell'angolo del robot rispetto alla verticale

Per l'acquisizione dei segnali viene sfruttato uno dei due convertitori analogico/digitale integrati nel microcontrollore, configurati in modo da eseguire il campionamento con il tempo di sample più lungo possibile (i segnali in oggetto non variano rapidamente, ma sono affetti da rumore) e da effettuare continuamente la scansione ciclica dei canali utilizzati.
Viene inoltre adoperato un canale DMA per effettuare il trasferimento del valore letto in una determinata area di memoria: ogni volta che il ciclo di conversione di tutti i canali viene terminato, un interrupt segnala al controller DMA che i dati sono disponibili per essere copiati nelle relative variabili, senza interrompere l'esecuzione del programma.
Per ridurre ulteriormente il rumore della misura viene utilizzata la tecnica dell'*oversampling*: è possibile aumentare la risoluzione di un convertitore analogico/digitale, a scapito della frequenza di conversione, mediando $N = 2^{2n}$ letture. Il numero n di bit guadagnati è quindi pari a:

$$n = \frac{log_2(N)}{2} \tag{6.1}$$

I dati convertiti dall'ADC vengono quindi salvati in una struttura di array, che vengono scritte direttamente dal controller DMA: è infatti possibile configurare l'ADC ed il canale DMA utilizzato per il tasferimento dei dati in maniera da eseguire N conversioni per K canali e quindi copiare in blocco i valori acquisiti in una struttura composta da K array di N celle ciascuno. Quando un componente del programma richiede il dato letto da un canale dell'ADC, una funzione apposita restituisce la media degli N campioni presenti nell'array relativo a quel canale.

I valori ricavati dall'acquisizione delle misure rese disponibili dall'accelerometro e dal giroscopio vengono utilizzati per stimare l'angolo del robot rispetto alla verticale mediante un filtro di Kalman (per la descrizione del funzionamento del filtro si rimanda alla Sezione 3.2.1).
Il codice che sintetizza il filtro di Kalman è stato ottimizzato manualmente al fine di permetterne l'esecuzione da parte del microcontrollore in tempi compatibili con il periodo del ciclo di controllo. In particolare le operazioni tra matrici sono state sviluppate in maniera da ridursi a cinque moltiplicazioni nella funzione di predict, mentre la routine che esegue l'update, in cui si effettuano trasposizioni e inversioni di matrici, è svolta mediante nove moltiplicazioni e due divisioni. Le operazioni di predict e di update del filtro vengono invocate all'interno del ciclo a 100 Hz e i valori di angolo e velocità angolare stimati vengono salvati nella struttura `Config` (6.1).

6.2.2 Segnali digitali

I segnali prodotti dagli encoder in quadratura montati sull'albero dei motori elettrici, utilizzati per ricavare i dati relativi all'odometria, sono acquisiti mediante dei timer opportunamente configurati. È infatti possibile impostare i timer in una modalità di funzionamento apposita per la decodifica dei segnali in quadratura generati dagli encoder incrementali.
In questa modalità due piedini del microcontrollore vengono impostati come ingresso flottante e connessi agli ingressi TI1 e TI2 di un timer. Il dispositivo è quindi utilizzato come un contatore, il cui valore viene aggiornato a seconda delle transizioni logiche di TI1 e TI2: ad ogni fronte di salita o di discesa il valore salvato nel registro del rispettivo timer viene incrementato o decrementato a seconda dello sfasamento tra i segnali in ingresso. Se ad esempio si definisce che il conteggio positivo sia per segnali su TI2 in ritardo rispetto a TI1, se il fronte si presenta prima su quest'ultimo il contatore verrà incrementato, altrimenti verrà decrementato. Un esempio di funzionamento del timer in modalità *encoder interface* è riportato in Figura 6.1.

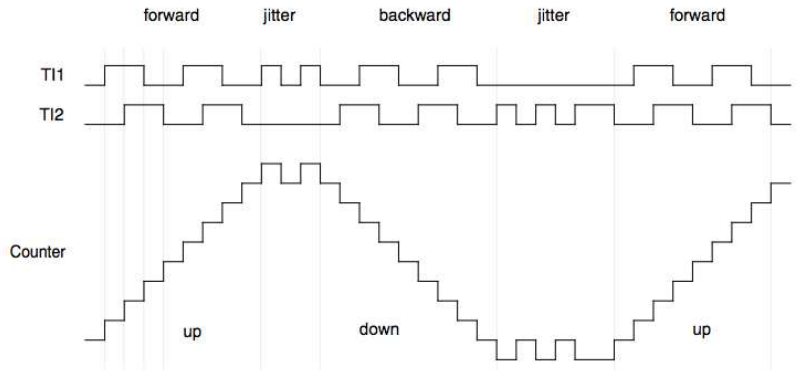


Figura 6.1: Acquisizione dei segnali in quadratura mediante timer

Il microcontrollore viene utilizzato anche per l'acquisizione dei segnali generati da interruttori a bordo del robot. Sono presenti, ad esempio, due interruttori collegati ai baffi di sicurezza che inibiscono il funzionamento dei motori, ed un tasto per l'azzeramento dell'offset di accelerometro e giroscopio, utilizzato per impostare lo zero dell'angolo del robot rispetto alla verticale.
Questo genere di segnali viene acquisito configurando il sistema di gestione degli interrupt in maniera che sia scatenato un evento ogni qual volta

si rilevi un fronte su uno dei piedini collegati agli interruttori. La routine richiamata effettua quindi il test sul piedino che ha scatenato l'interrupt e ne valuta lo stato, intraprendendo quindi l'operazione ad esso legata.

6.3 Controllo dei motori

Il software è predisposto per il controllo dei motori utilizzando due tecniche differenti, illustrate nella Sezione 5.3.1:

- sign/magnitude
- locked antiphase

Nella modalità sign/magnitude vengono utilizzati due segnali: il segnale DIR indica quale ramo del ponte H viene fatto commutare, e quindi il verso di rotazione del motore, mentre il segnale PWM è un'onda quadra che accende e spegne il ramo attivo del ponte H. Un segnale PWM con duty-cycle D nullo comporta che il motore sia fermo, mentre un segnale PWM con duty-cycle $D = 1$ corrisponde alla massima velocità di rotazione del motore.

Utilizzando la modalità locked antiphase entrambi i rami del ponte H sono attivi, e l'onda quadra viene utilizzata per invertire continuamente il verso di rotazione del motore. In questo caso, un segnale PWM con duty cycle $D = 0.5$ fa si che la corrente media che scorre nel motore sia nulla, ed esso sia quindi fermo, mentre un segnale con $D = 0$ oppure $D = 1$ comporta una rotazione alla massima velocità rispettivamente in una o nell'altra direzione.

Il microcontrollore si occupa di generare i segnali DIR e PWM per le due modalità di controllo, selezionabili impostando un parametro contenuto nella struttura `Config`. La generazione dell'onda quadra a duty cycle variabile è affidata ad uno dei timer disponibili, che funziona da contatore con fondoscala M. Ad ogni colpo di clock del timer, la cui frequenza è impostabile tramite appositi registi, il contatore viene incrementato, definendo quindi la frequenza dell'onda quadra generata. Ogni timer ha a disposizione 4 registri di comparazione $OC1 - 4$, che generano un evento quando il contatore ha lo stesso valore in essi contenuto, portando a livello logico alto il piedino di uscita corrispondente. Il conteggio procede fino a quando il contatore raggiunge il valore M, momento in cui il timer si azzera e i piedini

di uscita vengono riportati a livello logico basso. Si evince quindi che, per ogni canale, il duty cycle D vale:

$$D = \frac{OC}{M} \tag{6.2}$$

I motori sono controllabili sia ad anello aperto, impostando direttamente il valore del duty cycle D del segnale PWM, sia in anello chiuso per l'inseguimento di un determinato setpoint.
Il software prevede che sia possibile impostare setpoint di velocità, di accelerazione e di coppia.
Il valore impostato viene inseguito utilizzando dei controllori PID, che vengono attivati o disattivati dinamicamente a seconda del tipo di setpoint ricevuto.
Nel caso di setpoint di velocità e di accelerazione il segnale errore è ricavato confrontando il valore impostato con il dato fornito dall'encoder incrementale montato sull'albero del motore. Le costanti dei due controllori, di tipo PI, sono state ricavate mediante le regole di Ziegler-Nichols, discusse nella Sezione 7.1.2 e riportate in Tabella 7.1. Il controllo in coppia è invece realizzato utilizzando per ricavare il segnale errore il valore della corrente assorbita dai motori, misurata tramite le resistenze di sense (si veda la Sezione 5.3.3), da cui viene ricavato il valore della coppia esercitata mediante la relazione 5.14. In questo caso il controllore ha la sola componente integrale, meno sensibile ai picchi di corrente dovuti ai cambi di direzione che portano le componenti proporzionale e derivativa ad agire in maniera troppo brusca.

6.4 Interfaccia con l'esterno

Il software utilizza le periferiche di comunicazione seriale, sia su cavo che via rete wireless, per ricevere comandi dall'esterno e per inviare i dati ricevuti dai sensori e lo stato del robot ai dispositivi connessi.

La gestione dei comandi è realizzata tramite una *shell* in ascolto su una delle periferiche USART disponibili, tramite cui un utente, o un software esterno, può interagire col robot interrogandolo per ricevere il valore di un sensore o dello stato del sistema oppure può impartire dei comandi, ad esempio per attivare un particolare algoritmo di controllo o per forzare un valore particolare di PWM. I comandi messi a disposizione dal software permettono di gestire completamente il robot tramite un computer, potendo quindi

implementare sistemi di controllo anche esterni, come mostrato nella sezione successiva.

Il software può inoltre essere configurato per inviare periodicamente il valore di alcune variabili su una delle interfacce seriali, permettendo di registrare l'evolvere dei segnali su un dispositivo esterno. Sono state sviluppate alcune routine MATLAB che permettono di salvare i dati ricevuti o produrre dei grafici in tempo reale, al fine di facilitare le operazioni di taratura e debug degli algoritmi di controllo.

6.5 Algoritmi di controllo

I controllori progettati, descritti nel capitolo successivo, sono stati implementati sia a bordo del robot, eseguendo gli algoritmi sul microcontrollore, sia su un computer esterno, collegato al robot tramite interfaccia seriale.
In particolare, i controllori PID e LQR sono completamente eseguiti dal microcontrollore, che riceve i valori delle costanti tramite la shell oppure caricandoli dalla memoria flash interna. È possibile in ogni momento selezionare quale controllore utilizzare, e l'azione di controllo può essere generata anche sommando le azioni di diversi controllori: si può ad esempio scegliere di attivare il controllore PID che mantiene l'equilibrio del robot e il controllore PID che insegue un setpoint di velocità, oppure aggiungere del rumore all'uscita di un controllore per osservarne gli effetti. Quest'ultima possibilità è ad esempio sfruttata per produrre i dati su cui viene addestrato l'algoritmo di apprendimento per rinforzo, descritto nella Sezione 7.3.

Il controllore basato sul reinforcement learning richiede invece di poter accedere alla politica di controllo ottenuta, la cui dimensione dipende dal numero di variabili di stato e dalla discretizzazione degli stati scelta. Un problema di questo genere, in cui si hanno tre variabili di stato (angolo, velocità angolare e velocità lineare) e in cui è richiesta una buona risoluzione, fa si che le politiche generate siano troppo ingombranti per essere memorizzate all'interno della memoria del microcontrollore. Si è quindi scelto di utilizzare un software in esecuzione su un computer esterno che, ricevendo periodicamente le informazioni relative allo stato del robot, accede alla politica e sceglie l'azione da eseguire.

Capitolo 7

Controllo

"La vita è come andare in bicicletta: se vuoi stare in equilibrio devi muoverti."

Albert Einstein

Sono state studiate differenti tecniche per il controllo del moto del robot. Sono stati implementati due controllori classici, ovvero un regolatore PID che permette di mantenere il robot in equilibrio ed un controllore LQR tramite il quale è anche possibile inseguire un profilo di velocit 'a, e un controllore basato sul paradigma dell'apprendimento per rinforzo.
Nelle sezioni seguenti sono illustrate le caratteristiche di ciascun approccio e le relative applicazioni al problema del controllo del robot.

7.1 Controllore PID

7.1.1 Principio di funzionamento

I regolatori più usati in ambito industriale sono certamente i PID, sia perchè il loro impiego consente di controllare in modo soddisfacente un'ampia gamma di processi, sia grazie alla quantità di semplici regole per la loro taratura, applicabili anche quando non è disponibile un modello matematico preciso del sistema da controllare.
Nei regolatori PID la variabile di controllo u è generata come la somma di

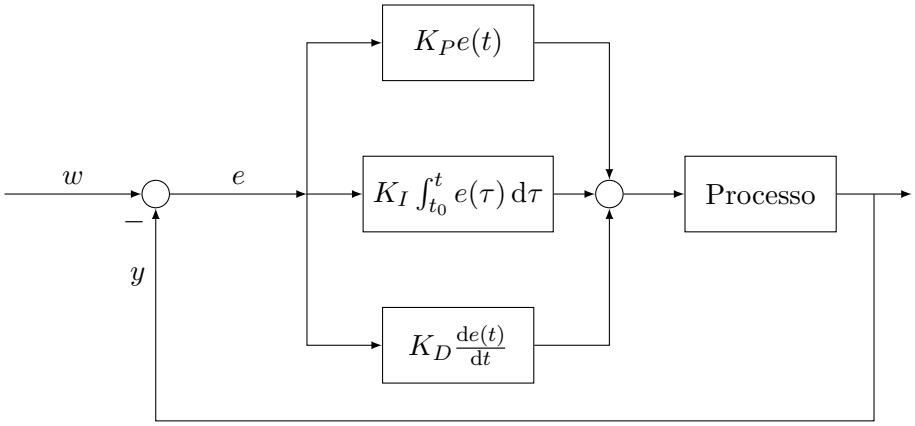


Figura 7.1: Schema a blocchi di un controllore PID

tre contributi. Il primo termine è proporzionale all'errore e tra il segnale di riferimento w e la variabile di uscita y del sistema sotto controllo. Il secondo contributo, proporzionale all'integrale di e (quindi al suo valor medio calcolato in una data finestra temporale), è necessario per imporre che l'errore si annulli asintoticamente a fronte di segnali di riferimento o disturbi additivi costanti. Il terzo contributo, proporzionale alla derivata di e, ha lo scopo di tentare di anticipare l'andamento dell'errore negli istanti futuri, applicando una correzione di tipo feed-forward.
L'andamento nel tempo della variabile di controllo è quindi:

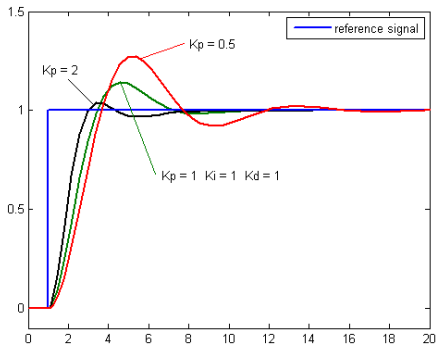
$$u(t) = K_P e(t) + K_I \int_{t_0}^{t} e(\tau)\,\mathrm{d}\tau + K_D \frac{\mathrm{d}e(t)}{\mathrm{d}t} \tag{7.1}$$

dove K_P, K_I e K_D sono costanti positive o nulle (nell'ipotesi che il guadagno del processo sia positivo).

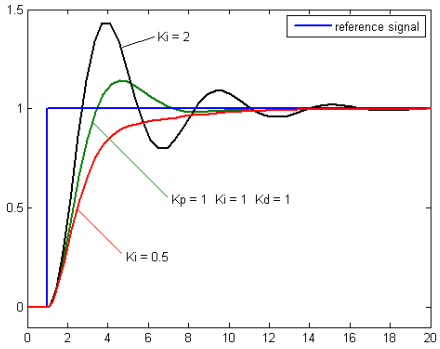
7.1.2 Taratura del controllore

Il contributo di ciascuna delle tre azioni alle prestazioni del controllore è analizzato in seguito, mostrando in Figura 7.2 i differenti effetti che la taratura delle costanti proporzionale, integrale e derivativa hanno sul comportamento dell'uscita.

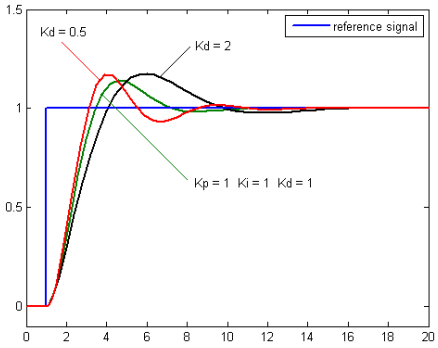
Il termine proporzionale, chiamato anche *guadagno* del controllore, ha effetto sulla risposta immediata del sistema ad un cambiamento del valore dell'errore. Il comportamento dell'uscita del controllore, a fronte di uno scalino in ingresso, al viariare di K_P, è mostrato in Figura 7.2(a). Un valore



(a) K_P varia, K_I e K_D costanti



(b) K_I varia, K_P e K_D costanti



(c) K_D varia, K_P e K_I costanti

Figura 7.2: Grafici della risposta allo scalino al variare delle costanti del controllore PID

troppo alto del guadagno può portare ad instabilità del sistema, generando un'oscillazione attorno al setpoint. Un valore di K_P troppo basso rende invece il controllore poco reattivo alle variazioni del segnale errore, fino al punto di produrre azioni troppo contenute perchè abbiano effetto sul sistema.

Il contributo del termine integrale, per la sua natura, è dovuto non solo all'ampiezza dell'errore ma anche alla sua durata nel tempo. Un errore, seppur di ampiezza contenuta, può restare costante nel tempo, andando a costituire un *offset*. Integrando questo valore si ottiene un termine che continua ad aumentare, permettendo al controllore di incrementare l'intensità della propria azione al fine di correggere la differenza tra uscita e setpoint. Il comportamento dell'uscita del controllore, a fronte di uno scalino in ingresso, al viariare di K_I, è mostrato in Figura 7.2(b). Il termine integrale aumenta quindi la velocità di risposta del sistema alle perturbazioni e permette di raggiungere, a regime, il setpoint. Essendo influenzata dall'errore accumulato negli istanti di tempo passati, la componente integrale può por-

tare ad una sovraelongazione dell'uscita. Un valore di K_I troppo elevato può quindi far comparire delle oscillazioni. Il termine integrale può essere causa di altri problemi, descritti nella Sezione 7.1.4.

Il termine derivativo ha effetto sulla velocità di variazione dell'uscita del controllore. Se l'errore sta aumentando, l'azione derivativa cerca di compensare questa deviazione in ragione della sua velocità di cambiamento, senza attendere che l'errore diventi significativo (azione proporzionale) o che persista per un certo tempo (azione integrale). Se al contrario l'errore sta diminuendo, la componente derivativa riduce l'azione del controllore al fine di limitare la sovraelongazione dell'uscita. Il comportamento dell'uscita del controllore, a fronte di uno scalino in ingresso, al viariare di K_P, è mostrato in Figura 7.2(c). Valori troppo alti di K_P possono rendere il controllore troppo nervoso, specie a fronte di perturbazioni rapide che hanno quindi derivata molto elevata, portando ad instabilità. Per questo motivo l'azione derivativa viene spesso tralasciata nelle implementazioni dei controllori PID.

Tra i vantaggi che hanno contribuito alla diffusione dei controllori PID ha sicuramente importanza la varietà di tecniche a cui si può ricorrere per la loro taratura. Ai metodi analitici, basati sullo studio del modello del sistema e sulla ricerca dei parametri ottimi per il controllo del processo, si affiancano alcuni metodi pratici che permettono di eseguire la taratura modificando le costanti del controllore ed osservando il differente comportamento dell'uscita.
Uno dei metodi più utilizzati, grazie alla semplicità della procedura e alla possibilità di raggiungere buoni risultati in tempi molto rapidi, è il metodo di Ziegler-Nichols.

Il procedimento prevede di seguire alcuni passi mentre si osserva l'uscita del sistema, e di tarare quindi il sistema utilizzando semplici relazioni:

- il processo viene inizialmente controllato da un controllore esclusivamente proporzionale (K_I e K_D poste a zero)
- il guadagno K_P del controllore proporzionale viene gradualmente aumentato, osservando il comportamento dell'uscita
- quando il valore dell'uscita comincia a presentare delle oscillazioni sostenute, si registra il guadagno critico K_C per il quale sono insorte le oscillazioni e si misura il periodo dell'oscillazione T_C
- utilizzando le relazioni riportate in Tabella 7.1 si determinano le costanti per la taratura del controllore di tipo P, PI oppure PID

Tabella 7.1: Regole di Ziegler-Nichols

Tipo di controllore	K_P	K_I	K_D
P	$0.5K_C$	-	-
PI	$0.45K_C$	$1.2K_C/P_C$	-
PID	$0.6K_C$	$2K_C/P_C$	-

7.1.3 Controllore PID digitale

L'implementazione di un controllore PID in un sistema digitale richiede il passaggio dal dominio tempo continuo al dominio tempo discreto. La trasformata di Laplace della funzione di trasferimento è

$$\frac{U(s)}{E(s)} = K_P + \frac{K_I}{s} + K_D s \tag{7.2}$$

Applicando il metodo della *backward difference* si può ricavare il controllore equivalente discreto:

$$s = \frac{1 - z^{-1}}{T_S} \tag{7.3}$$

$$u(n) = K_P e(n) + K_I e(n) \frac{T_S}{1 - z^{-1}} + K_D e(n) \frac{1 - z^{-1}}{T_S} \tag{7.4}$$

Semplificando si ottiene:

$$\begin{aligned} u(n) = {} & u(n-1) + K_P[e(n) - e(n-1)] + K_I T_S e(n) \\ & + \frac{K_D}{T_S}[e(n) - 2e(n-1) + e(n-2)] \end{aligned} \tag{7.5}$$

Questa forma può essere facilmente tradotta in codice eseguibile sul microcontrollore.

7.1.4 Implementazione

Il controllore è stato implementato utilizzando come segnale errore la differenza tra l'angolo di inclinazione del telaio e la verticale, al fine di mantenere l'equilibrio. Sono state utilizzate le sole componenti proporzionale e integrativa: la prima permette una reazione rapida del robot alle perturbazioni, mentre la seconda è necessaria per garantire il raggiungimento del setpoint

a regime. La componente derivativa è invece stata disattivata, in quanto induceva comportamenti nervosi che peggioravano le prestazioni ottenute.
La taratura delle costanti è stata effettuata seguendo le regole di Ziegler-Nichols, mostrate in Tabella 7.1. I controllori PID sono affetti da alcune problematiche, tra le quali le più ricorrenti sono legate al fenomeno di saturazione dell'azione integrale, conosciuto come *wind-up*.
L'insorgenza del windup può verificarsi quando il comando di attuazione è vincolato a dei valori di saturazione; in tal caso supponendo di applicare al sistema di riferimento un disturbo a gradino, l'integratore inizierà ad accrescere il suo valore in uscita per via dell'errore non nullo. In caso di saturazione, il comando di attuazione resterà costante anche se l'uscita dell'integratore continuerà a crescere fino a quando l'errore non diventerà nullo. Se il sistema è stabile, dopo un certo intervallo di tempo, l'errore diventerà nullo e l'integratore inizierà a scaricarsi e fino a quando il valore di uscita non sarà inferiore alla saturazione (relativa all'attuatore), il segnale di controllo rimarrà costante. Questo fenomeno fa si che il sistema dopo aver raggiunto la condizione di errore nullo, si allontani in direzione opposta, creando un effetto di sovraelongazione dalle caratteristiche non lineari. Per limitare questo problema sono state apportate le seguenti modifiche al controllore PID classico:

- la finestra temporale su cui viene integrato l'errore è limitata
- l'errore accumulato dall'integratore è limitato ad un valore massimo e minimo
- se l'errore e cambia segno il valore accumulato dall'integratore viene azzerato

7.2 Controllore LQR

7.2.1 Teoria del controllo ottimo e controllore LQR

Il controllore LQR, dall'inglese *Linear Quadratic Regulator*, fa parte degli algoritmi di controllo inseriti nell'ambito della teoria del controllo ottimo. Fanno parte di questa categoria quei controllori in grado di stabilizzare un sistema dinamico minimizzando una particolare cifra di merito, dipendente dallo stato del sistema e dal vettore degli ingressi [18].

Un generico sistema dinamico a tempo continuo è descritto da equazioni del tipo:

$$\dot{x}(t) = f(x(t), u(t), t) y(t) = g(x(t), u(t), t) \tag{7.6}$$

Se le funzioni f e g non dipendono esplicitamente dal tempo il sistema si dice *invariante nel tempo*, o *stazionario*. Se inoltre le funzioni f e g sono lineari in x e y, ovvero sia $\dot{x}(t)$ che $y(t)$ sono combinazioni lineari delle varie componenti dei vettori $x(t)$ e $u(t)$, allora il sistema si dice sistema lineare tempo invariante (sistema LTI) e può essere rappresentato nella forma:

$$\dot{x}(t) = Ax(t) + Bu(t) y(t) = Cx(t) + Du(t) \tag{7.7}$$

Il problema del controllo di un pendolo inverso tratta un sistema non lineare, che quindi deve essere linearizzato nell'intorno di un punto di interesse per poter essere ridotto ad un sistema LTI.
Il procedimento della linearizzazione prevede di ricavare le matrici A, B, C e D mediante il calcolo delle derivate parziali delle funzioni f e g rispetto alle variabili x e u, nell'intorno del punto di equilibrio:

$$\begin{aligned} A(t) &= \frac{\partial f(x,u)}{\partial x}\Big|_{x=\tilde{x}(t),u=\tilde{u}(t)} & B(t) &= \frac{\partial f(x,u)}{\partial u}\Big|_{x=\tilde{x}(t),u=\tilde{u}(t)} \\ C(t) &= \frac{\partial g(x,u)}{\partial x}\Big|_{x=\tilde{x}(t),u=\tilde{u}(t)} & D(t) &= \frac{\partial g(x,u)}{\partial u}\Big|_{x=\tilde{x}(t),u=\tilde{u}(t)} \end{aligned} \tag{7.8}$$

Dato un sistema LTI, il controllore LQR permette di ottenere un controllo in retroazione ottimo rispetto ad una funzione di costo quadratica dello stato $x(t)$ e del controllo $u(t)$ [19]. Nella forma tempo continua ad orizzonte finito, la funzione di costo J è definita come:

$$J = \frac{1}{2} x^T(T) F(T) x(T) + \int_0^T \left(x^T Q x + u^T R u \right) dt \tag{7.9}$$

in cui Q e S sono simmetriche e semidefinite positive, mentre R è simmetrica e definita positiva.
La matrice Q definisce quanto sia il peso di ciascuna variabile di controllo all'interno della funzione costo. Ad esempio, una matrice del tipo:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 10 \end{bmatrix} \tag{7.10}$$

da molta più priorità al contenimento del valore della seconda variabile di stato, pesata 100, rispetto alle altre due.
La legge di controllo che minimizza il valore della funzione costo è:

$$u = -Kx \tag{7.11}$$

in cui K è data da:

$$K = R^{-1}B^T P(t) \tag{7.12}$$

e P viene ricavata mediante la soluzione dell'equazione di Riccati

$$A^T P(t) + P(t)A - P(t)BR^{-1}B^T P(t) + Q = -\dot{P}(t) \tag{7.13}$$

7.2.2 Implementazione

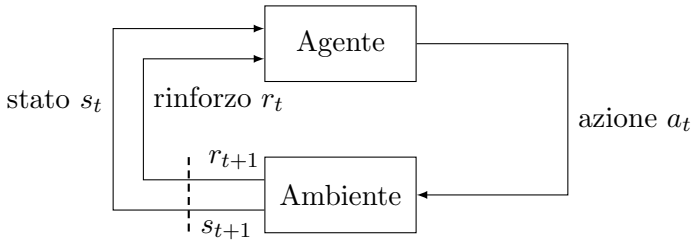
Nel caso in esame il controllore LQR riceve in ingresso quattro variabili di stato: l'angolo del telaio rispetto alla verticale, la velocità angolare, la posizione rispetto ad un riferimento e la velocità di spostamento. La variabile di controllo è la coppia richiesta ai motori al fine di raggiungere il setpoint imposto.
Il controllore LQR è stato sintetizzato sulla base delle simulazioni effettuate in ambiente MATLAB/Simulink. Con gli strumenti che questo pacchetto mette a disposizione si è potuto tarare il controllore dopo aver linearizzato il sistema nell'intorno del punto di equilibrio instabile e aver definito i pesi da dare alle variabili di stato, ovvero la matrice Q.
Utilizzando il pacchetto *Control System Toolbox*, anch'esso facente parte dell'ambiente MATLAB, si può quindi ricavare la legge di controllo ottima, ovvero si determina la matrice K.
Il controllore ricavato è stato quindi implementato a bordo del robot al fine di controllare la stabilità del robot e di inseguire un setpoint di velocità impostato dall'esterno (nella Sezione 8.4 sono riportati i risultati ottenuti).

7.3 Controllo basato su apprendimento per rinforzo

7.3.1 Apprendimento per rinforzo

L'apprendimento per rinforzo è una paradigma per l'identificazione di politiche di controllo sviluppato nell'ambito dell'intelligenza artificiale, e più pre-

cisamente nel campo dell'apprendimento automatico, che punta a realizzare algoritmi in grado di apprendere dall'esperienza accumulata e di adattarsi alle mutazioni dell'ambiente. Questa tecnica di programmazione si basa sul presupposto di potere ricevere degli stimoli dall'esterno in seguito alle scelte effettuate dall'algoritmo, che possono rivelarsi più o meno utili. Viene definito quindi un premio in caso sia stata scelta un'azione corretta, che avvicina all'obiettivo, mentre una scelta errata comporta una penalizzazione del sistema.
L'obiettivo perseguito dal sistema è il raggiungimento del maggior premio possibile e di conseguenza del migliore risultato possibile.
Quando si affronta un problema utilizzando le tecniche di RL (dal termine inglese *Reinforcement Learning*) ci si chiede quindi quali azioni, tra quelle possibili, i soggetti principali (gli agenti) devono intraprendere nell'ambiente in cui operano, in modo da massimizzare la ricompensa nel lungo periodo (rinforzo atteso). Lo scopo degli algoritmi di RL è far sviluppare all'agente un comportamento (politica) che lo porti a selezionare le azioni più adatte a raggiungere un determinato obiettivo.



Si definisce *agente* un'entità in grado di percepire l'ambiente che lo circonda (ad esempio tramite dei sensori) e di eseguire delle azioni che, interagendo con esso, modificano lo stato del sistema (ad esempio tramite degli attuatori). È definito *ambiente* quindi l'insieme degli elementi che possono partecipare ad un qualche tipo di interazione con l'agente.
La tecnica di apprendimento per rinforzo formalizza le interazioni tra agente ed ambiente permettendo di definire un metodo per selezionare, in ogni stato del sistema, l'azione da eseguire.
L'interazione tra agente e ambiente è continua: quando l'agente esegue delle azioni riceve in risposta dall'ambiente una descrizione, eventualmente parziale, dello stato del sistema modificato dall'azione eseguita e un *rinforzo*, ovvero un valore numerico che rappresenta l'utilità dell'azione eseguita al fine di raggiungere l'obiettivo.

Il paradigma del Reinforcement Learning prevede che, data una sequen-

za discreta di istanti temporali, l'agente interagisca con l'ambiente in ogni istante ricevendo una rappresentazione dello stato del sistema $s_t \in S$, dove S è l'insieme di tutti i possibili stati in cui il sistema può trovarsi. Sulla base di questa rappresentazione, l'agente seleziona un'azione a_t nell'insieme $A(s_t)$, ovvero l'insieme delle azioni che l'agente può intraprendere se si trova nello stato s_t. All'istante temporale successivo l'agente riceverà, come conseguenza dell'azione eseguita, un valore numerico r_{t+1} tra quelli appartenenti all'insieme dei rinforzi R, e si sposterà nello stato successivo s_{t+1}. Il rinforzo rappresenta la misura della prestazione del sistema da massimizzare nel tempo.
Ad ogni istante temporale, l'agente, con una certa probabilità, esegue un'azione che lo porta nello stato successivo, costruendo una sequenza di azioni che prende il nome di *politica*, indicata con il simbolo π, dove $\pi_t(s, a)$ indica la probabilità che $a_t = a$ se $s_t = s$. L'obiettivo dell'agente è scegliere una politica tale da massimizzare la somma totale dei rinforzi che riceverà eseguendo la sequenza di azioni pianificata.
Per assegnare un compito all'agente bisogna quindi specificare dei valori del rinforzo tali da premiarlo in caso si stia avvicinando all'obiettivo e punirlo se le azioni eseguite siano controproducenti. In questo senso il segnale di rinforzo è il modo di comunicare all'agente cosa si vuole che esso faccia, lascandogli il compito di scegliere il modo in cui farlo.
La somma dei rinforzi ottenuti è naturalmente nota solo a posteriori, motivo per cui viene utilizzato il concetto di *rinforzo atteso* R, che indica la somma dei rinforzi che l'agente si aspetta di ottenere intraprendendo un certo piano per il raggiungimento dell'obiettivo. Nel caso di problemi continui, in cui le interazioni tra agente e ambiente si susseguono ininterrottamente, la somma dei rinforzi non può essere eseguita su un numero infinito di lavori. È quindi necessario introdurre un tasso di sconto, indicato con γ, tramite cui si definisce il *rinforzo scontato*:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + K = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{7.14}$$

Il tasso di sconto γ è compreso tra 0 e 1 ed indica il valore attuale dei futuri rinforzi. Se il valore dello sconto è minore di 1 la somma infinita ha valore finito. Se lo sconto è pari a 0, si dice che l'agente è *miope*, perchè il rinforzo atteso viene a coincidere col rinforzo dell'istante successivo al t-esimo, ovvero il sistema è concentrato solo nel massimizzare i rinforzi immediati.

Gran parte degli algoritmi di RL sono basati sulla stima della *funzione di utilità* (dall'inglese *Value Function*), una funzione degli stati (o delle coppie

stato-azione) che stima quanto sia vantaggioso per l'agente trovarsi in un determinato stato, o quanto convenga scegliere una determinata azione nel caso ci si trovi in quello stato.
Concettualmente, la funzione di utilità di uno stato s applicando una politica π viene indicata con $V^\pi(s)$ ed è il ritorno atteso nell'ipotesi che dallo stato s e dall'istante di tempo t in cui il sistema si trova viene rispettata π. Indicando con E_π il valore atteso, $V^\pi(s)$ è definita come:

$$V^\pi(s) = E_\pi R_t | s_t = s = E_\pi \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \tag{7.15}$$

Analogamente si può definire la funzione di utilità di un'azione a nello stato s, indicata con $Q^\pi(s,a)$, supponendo di eseguire l'azione a e, dall'istante t, di applicare la politica π:

$$Q^\pi(s,a) = E_\pi R_t | s_t = s, a_t = a = E_\pi \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a \tag{7.16}$$

Risolvere un problema utilizzando le tecniche di RL significa trovare una politica che consenta di ottenere il massimo rinforzo a lungo termine. La funzione di utilità permette di stabilire un parziale ordinamento tra le diverse politiche applicabili. Per definizione, una politica π è considerata migliore di un'altra politica π' se il suo ritorno atteso è maggiore a quello di π' per tutti gli stati.
Formalmente:

$$\pi \geq \pi' \iff V^\pi(s) \geq V^{\pi'}(s) \forall s \in S \tag{7.17}$$

Per ogni problema esiste sempre una (o piu' d'una) politica migliore delle altre, che viene chiamata *politica ottima* ed è indicata col simbolo π^*. Le politiche ottime condividono le stesse funzioni di utilità di stato e di azione, chiamate rispettivamente *funzione di utilità di stato ottima* e *funzione di utilità di azione ottima*, cosi' definite:

$$V^*(s) = max_\pi V^\pi(s) Q^*(s,a) = max_\pi Q^\pi(s,a) \tag{7.18}$$

7.3.2 Q-learning

Tra gli approcci sviluppati per risolvere problemi di apprendimento per rinforzo uno dei più utilizzati è l'algoritmo di *Q-Learning*. Il Q-Learning è un algoritmo di *Temporal-Difference learning* (*TD*), ovvero un metodo predittivo che unisce idee del *Metodo Monte Carlo* e caratteristiche delle tecniche

di *Programmazione Dinamica* (*DP* dall'inglese *Dynamic Programming*) [20]. Gli algoritmi di TD sono simili al Metodo di Monte Carlo in quanto apprendono osservando le variazioni dell'ambiente in seguito all'applicazione di una determinata politica; ricordano anche le tecniche di Programmazione Dinamica perchè stimano il rinforzo ricorrendo ad approssimazioni successive, come mostrato in seguito.
L'algoritmo di Q-Learning viene anche detto *off-policy*, ovvero indipendente dalla politica, per la caratteristica di poter utilizzare una politica per interagire con l'ambiente (ad esempio una politica molto aleatoria, per favorire l'esplorazione), chiamata *behavior policy*, mentre viene stimata la funzione di utilità per un'altra politica (*estimation policy*).
L'algoritmo cerca di stimare, per ogni stato s, o per ogni coppia stato-azione (s, a), un valore che sia direttamente correlato al rinforzo totale che può essere ottenuto a partire da quello stato. La stima viene aggiornata utilizzando le tecniche di Programmazione Dinamica, secondo il modello

$$\begin{aligned}\texttt{nuova approssimazione} = \texttt{approssimazione precedente} + \\ \alpha(\texttt{nuova stima} - \texttt{approssimazione precedente})\end{aligned}$$

dove il parametro α, detto velocità di apprendimento (*learning rate*), determina la proporzione con cui la nuova stima della funzione di utilità e la precedente approssimazione vengono combinate per ottenere il nuovo dato. Nella sua forma più semplice, l'algoritmo Q-Learning si riduce ad un'unica regola di aggiornamento (*one-step Q-Learning*):

$$Q_{t+1}(s,a) \leftarrow \underbrace{Q_t(s,a)}_{\textit{old value}} + \underbrace{\alpha_t}_{\substack{\textit{learning}\\ \textit{rate}}} \times [\overbrace{\underbrace{r_t}_{\textit{reward}} + \underbrace{\gamma}_{\substack{\textit{discount}\\ \textit{factor}}} \underbrace{\max_{a'} Q_t(s',a')}_{\textit{max future value}}}^{\textit{expected discounted reward}} - \overbrace{Q_t(s,a)}^{\textit{old value}}] \tag{7.19}$$

7.3.3 Apprendimento *offline*: fitted Q-iteration

L'applicazione dell'algoritmo di Q-Learning a problemi reali, generalmente caratterizzati da spazio degli stati e spazio delle azioni molto vasti, introduce evidentemente la difficoltà nel calcolare il valore della funzione di utilità per ogni coppia stato-azione. Nel caso in cui gli spazi abbiano dimensioni continue, come tipicamente accade per lo spazio degli stati in un ambiente reale, all'enorme onere computazionale si somma l'impossibilità di esplorare tutte le coppie stato-azione osservabili. Per poter superare queste limitazioni, il

sistema deve essere in grado di *generalizzare* [21], ovvero avere la capacità di predire correttamente nuove informazioni mai raccolte durante la fase di addestramento. In questo modo l'esperienza acquisita su un sottoinsieme limitato dello spazio degli stati può essere opportunamente generalizzata per produrre una buona approssimazione che valga per un insieme di stati molto piu' ampio.
Questo approccio è tipico dell'apprendimento supervisionato, un altra branca dell'apprendimento automatico. Le tecniche di apprendimento supervisionato vengono utilizzate per generare una funzione in grado di descrivere sinteticamente un problema mediante l'analisi di dati di training. Il processo di apprendimento supervisionato prevede due fasi distinte:

fase di addestramento (o *learning*), in cui al sistema viene fornito un insieme di coppie input-output tramite le quali viene stimata una funzione h' in grado di classificare correttamente le coppie fornite

fase di generalizzazione , in cui al sistema viene fornito un diverso insieme di input (di cui si conosce l'output) e si valuta l'accuratezza della funzione stimata in base al numero di risposte corrette

Gli algoritmi di apprendimento supervisionato si basano sul presupposto che, se viene fornito un numero adeguato di dati d'esempio, è possibile generare una funzione h' in grado di approssimare la funzione h che descrive il sistema. Più in generale, la tecnica descritta si riconduce all'*approssimazione di funzioni*, in quanto permette di estrarre nuove informazioni da una funzione parziale, ricavata mediante l'analisi di un insieme ristretto di dati, cercando di generalizzarla per costruire un'approssimazione dell'intera funzione.
Nell'ambito dell'apprendimento per rinforzo la funzione in esame è la funzione utilità che, nel caso di spazio degli stati continuo, necessita di un'approssimazione in modo da poter essere rappresentata in una forma compatta e molto più generale. In questo contesto però l'insieme dei dati di training non è noto a priori, in quanto l'output dipende dalla funzione approssimata stessa. Nasce quindi l'esigenza di sfruttare le diverse tecniche di apprendimento per implementare un metodo che permetta di utilizzare algoritmi tipici dell'apprendimento supervisionato al fine di creare un'approssimazione della funzione di utilità Q, tipica dell'approccio RL, mediante elaborazione *offline* di un insieme di dati d'esempio raccolti dal sistema in analisi, sintetizzando successivamente una politica di controllo basata sulla funzione Q ricavata. Sulla base di queste considerazioni è stato sviluppato l'algoritmo *fitted Q-iteration*[21] [22].

L'algoritmo fitted Q-iteration (o FQI) è un metodo di apprendimento per rinforzo iterativo, che si basa sulla risoluzione di una sequenza di problemi di apprendimento supervisionato. L'algoritmo, a differenza dei classici metodi RL online, non interagisce direttamente con il sistema (*batch-mode Reinforcement Learning*) determinando una politica di controllo, che approssima il più possibile la politica ottima, eseguendo un'analisi offline di un insieme di informazioni chiamato *dataset*. Un dataset è composto da una serie di quadruple di forma (s_t, a_t, s_{t+1}, r_t) in cui s_t indica lo stato attuale, a_t l'azione intrapresa, s_{t+1} lo stato in cui il sistema si porta e r_t il rinforzo immediato ottenuto.

Alla prima iterazione l'algoritmo viene utilizzato per stimare il valore atteso del rinforzo, ovvero:

$$Q_1(s, a) = E_\pi R_t | s_t = s, a_t = a \tag{7.20}$$

Ad ogni passo successivo l'algoritmo utilizza l'intero dataset assieme alla funzione calcolata al passo precedente per determinare un nuovo insieme di dati di training utilizzato da un sistema di apprendimento supervisionato, chiamato *regressore*, per calcolare il successivo valore della funzione. In questo modo, ad ogni iterazione viene aumentato l'orizzonte temporale, fino a quando si raggiungono le condizioni di arresto. In seguito è mostrato lo pseudo codice che descrive l'algoritmo FQI.

Algorithm 1 Algoritmo fitted Q-iteration

repeat

$N \leftarrow N + 1$

creare un nuovo dataset $\mathcal{TS} = (i^l, o^l), l = 1, \ldots, \sharp\mathcal{D}$ utilizzando l'ultima stima Q_{N-1} e il dataset di partenza $\mathcal{D}$, in cui:

$$i^l = (s_t^l, a_t^l) \tag{7.21}$$

$$o^l = r_t^l + \gamma \max_{a'} Q_{N-1}(s_{t+1}^l, a') \tag{7.22}$$

applicare l'algoritmo di regressione R al dataset $\mathcal{TS}$ per approssimare la nuova funzione $Q_N(s_{t+1}, a)$

until una condizione di arresto viene raggiunta

return $\pi_N(s) = argmax_{a \in A} Q_N(s, a)$

Le condizioni di arresto possono essere diverse. Si può dichiarare a priori il numero massimo di iterazioni da eseguire, definendo quindi l'orizzonte

temporale, oppure calcolare ad ogni passo la distanza tra l'ultima approssimazione ricavata (Q_N) e quella precedente (Q_{N-1}), terminando il ciclo quando la differenza raggiunge un valore di soglia minimo.

Quando vengono raggiunte le condizioni di arresto la politica di controllo ottenuta è definita come:

$$\pi_N(s) = argmax_{a \in A} Q_N(s, a) \tag{7.23}$$

L'algoritmo fitted Q-iteration gode delle seguenti proprietà:

- è *svincolato dal modello*: l'unica informazione necessaria è un insieme di dati sotto forma di quadruple (stato, azione, stato successivo, rinforzo)
- è *legato alla bontà dei dati*: più i dati raccolti riescono a descrivere in maniera efficiente il sistema, più i risultati prodotti dall'algoritmo saranno buoni. È possibile apprendere politiche di successo anche partendo da un insieme ridotto di dati
- è *in grado di generare politiche di controllo* con prestazioni paragonabili a quelle ottenute tramite i controllori analitici, nonostante sia necessaria molta meno conoscenza del sistema rispetto ad essi

7.3.4 Regressore Extra-Trees

Nell'ambito dell'apprendimento supervisionato sono state sviluppate numerose tecniche di regressione, alcune delle quali sfruttano le strutture ad albero (metodi *tree-based*). Gli algoritmi basati sugli alberi di regressione ripartiscono l'insieme degli input in regioni distinte, assegnando a ciascuna di esse una predizione del valore di uscita, che viene calcolato come media tra gli output presenti nel dataset $\mathcal{TS}$ che appartengono a quella determinata regione. Il modello prodotto dai metodi di regressione ad alberi è ottenuto mediando le stime di differenti alberi, generati compiendo scelte differenti durante la classificazione dei dati.

Il metodo *Extra-Trees*, abbreviazione di extremely randomized trees [23], genera M alberi sfruttando per intero le informazioni presenti nel dataset $\mathcal{TS}$, invece che iniziare l'analisi da un determinato campione come altri metodi quali KD-Tree [21] e Tree Bagging [24].

La creazione degli Extra-Trees dipende da tre parametri:

M definisce il numero di alberi che vengono generati: gli studi effettuati nell'ambito dei metodi che sfruttano la casualità hanno reso noto che l'errore di predizione decresce in funzione di questo parametro, portando quindi a risultati più accurati al crescere di M

K indica quanti *split* (suddivisioni delle regioni in regioni più piccole) vengono generati ad ogni iterazione dell'algoritmo. Il valore di K è compreso tra 0 ed n, ed indica quante sono le dimensioni lungo cui effettuare il taglio casuale. Se $K = 1$ si ha una scelta completamente casuale (*totally randomized trees*), mentre se $K = n$ si effettueranno n tagli, in seguito confrontati per mantenere soltanto quello che ha ridotto maggiormente la varianza delle dei dati appartenenti alle foglie ottenute rispetto alla foglia di partenza

n_{min} indica il minimo numero di elementi che possono comporre una foglia. Valori elevati di n_min portano alla costruzione di alberi poco profondi, con valori di distorsione molto elevati e di varianza molto bassi, mentre il valore minimo $n_min = 2$ genera alberi molto profondi con un incremento sostanziale del numero di nodi. Il valore ottimo di n_min dipende fortemente dal rumore dei dati di output, motivo per cui è necessario valutare per ogni situazione quale sia la scelta migliore

7.3.5 Implementazione

Le tecniche di controllo basate sull'apprendimento per rinforzo mostrate in questo capitolo sono state applicate al problema in questione sia per ricavare un modello del robot sia per stimare una politica di controllo valida.
Il procedimento adottato consiste dapprima nel far eseguire al robot azioni casuali, registrando i valori relativi allo stato del robot e le azioni intraprese. Sulla base delle informazioni raccolte è stato prodotto il dataset per l'apprendimento offline, che consiste in un insieme di dati del tipo stato precedente - azione effettuata - stato successivo, utilizzati dall'algoritmo di Temporal Difference per stimare il modello del robot.

La politica di controllo generata dipende dal rinforzo che si attribuisce ai differenti cambiamenti di stato. Se l'obiettivo è mantenere l'equilibrio, si può imporre un rinforzo negativo pari all'inclinazione del robot:

$$r = -\alpha \tag{7.24}$$

in maniera che l'algoritmo scelga le azioni che lo conducano ad avere rinforzo massimo ovvero angolo pari a zero.
Per cercare di ridurre le oscillazioni attorno alla verticale si può imporre che il rinforzo sia pari a zero all'interno di un range di valori:

$$r = \begin{cases} 0 & \text{se} |\alpha| \leq \bar{\alpha} \\ -\alpha & \text{se} |\alpha| > \bar{\alpha} \end{cases} \tag{7.25}$$

Se l'obettivo, oltre a mantenere l'equilibrio, è anche far inseguire al robot un setpoint di posizione o di velocità, sarà necessario costruire un rinforzo r che tenga conto anche dell'errore di posizione o dell'errore di velocità. Dando peso maggiore o minore alle varie componenti del rinforzo si ottiene una politica più volta a raggiungere il setpoint di una variabile di stato piuttosto che di un'altra.
Si rimanda alla Sezione 8.5 per l'analisi degli esperimenti svolti e dei risultati ottenuti.

Capitolo 8

Realizzazioni sperimentali e valutazione

"Alle volte uno si crede incompleto ed è soltanto giovane."

Il visconte dimezzato - Italo Calvino

Nelle sezioni seguenti sono descritte le prove sperimentali effettuate e i risultati a cui esse hanno portato.
I test svolti hanno riguardato inizialmente la valutazione della qualità dell'inclinazione stimata rispetto al dato reale e la quantificazione del ritardo dei segnali rilevati e del movimento del robot rispetto all'applicazione dei comandi.
Successivamente sono riportati i comportamenti dei differenti algoritmi di controllo, evidenziandone gli aspetti positivi ed analizzandone le problematiche.

8.1 Stima dell'angolo del robot

La stima dell'angolo del robot rispetto alla verticale viene effettuata utilizzando i dati provenienti dall'accelerometro e dal giroscopio, opportunamente trattati tramite un filtro di Kalman (presentato nella Sezione 3.2.1).

Il filtro è stato inizialmente tarato sulla base dei dati forniti dal costruttore relativamente al rumore dei sensori e alla loro precisione, ricavando dei parametri che sono poi stati ritoccati osservando i risultati delle prove sperimentali.
Le prove sono state eseguite utilizzando un encoder ottico montato su un'asta collegata al telaio del robot e permanentemente in contatto col terreno, che fornisce il dato reale dell'inclinazione con cui confrontare le stime ottenute tramite il filtraggio.
In seguito sono mostrati i risultati ottenuti dopo il tuning dei parametri del filtro.

I grafici mostrati in Figura 8.1 mettono a confronto l'angolo stimato dal filtro di Kalman con il dato reale, registrati mantenendo spenti i motori del robot e muovendo il telaio manualmente, facendogli compiere oscillazioni di diverse ampiezze.
Le prove effettuate hanno mostrato che per ridurre il ritardo della stima rispetto all'angolo reale e per ottenere una buona immunità ai disturbi provocati dai movimenti lineari del robot è necessario pesare maggiormente il dato fornito dal giroscopio, utilizzato nella fase di predict, rispetto al dato fornito dall'accelerometro, utilizzato per l'update.
Si può notare che l'angolo stimato da Kalman insegue molto bene l'inclinazione reale, senza presentare ritardo e mostrando una buona accuratezza.
L'effetto della scelta dei differenti pesi legati ai due sensori si nota nelle oscillazioni di ampiezza maggiore: la fase di predict, che si basa sul valore della velocità angolare per predirre lo stato successivo, porta a sovrastimare l'angolo, che non viene immediatamente corretto proprio a causa del minor peso dato al valore letto dall'accelerometro durante la fase di update. Questo fenomeno non rappresenta comunque un problema, in quanto le condizioni in cui la sovrastima è più evidente si presentano difficilmente quando il robot è governato da un controllore, che limiterà i movimenti del telaio a piccole oscillazioni attorno al setpoint.

In Figura 8.2 sono mostrati i risultati ottenuti controllando il robot con un comando PWM casuale, che produce rapidi cambi di direzione e oscillazioni molto variabili.
Anche in queste condizioni la stima dell'angolo si mantiene fedele al dato reale, anticipando quest'ultimo nelle situazioni in cui la velocità angolare diminuisce bruscamente per il fenomeno della sovrastima descritto in precedenza.

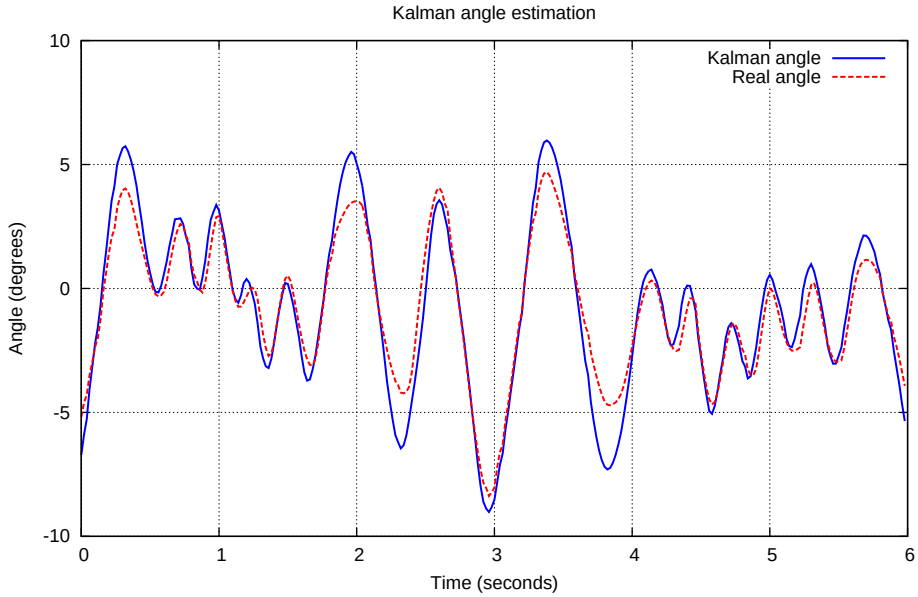


Figura 8.1: Confronto tra inclinazione reale e stima tramite filtro di Kalman: robot mosso manualmente

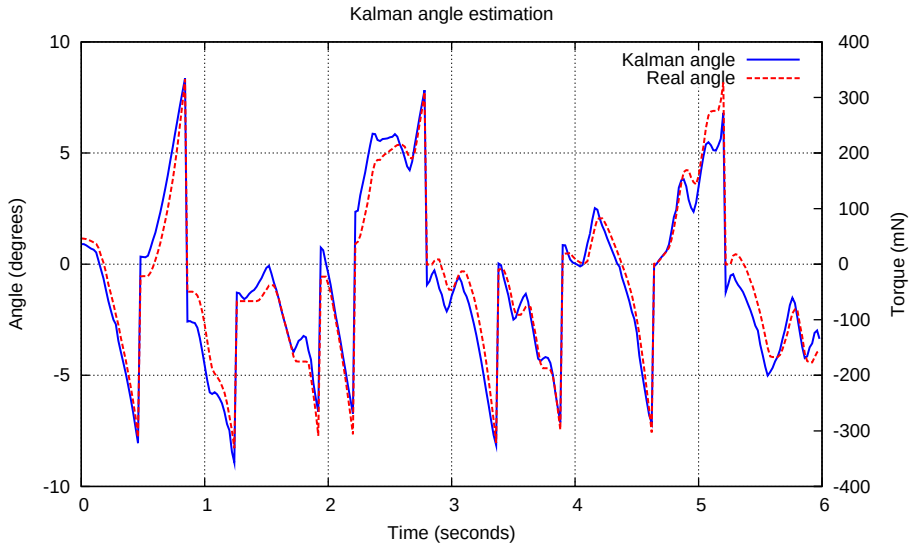


Figura 8.2: Confronto tra inclinazione reale e stima tramite filtro di Kalman: robot controllato imponendo un comando casuale ai motori

8.2 Ritardo dei segnali e test di crosscorrelazione

Al fine di valutare la qualità dei dati ricavati dai sensori sono stati eseguiti dei test di crosscorrelazione tra essi e i dati reali, per evidenziare eventuali ritardi. La necessità di effettuare un'indagine di questo tipo si è presentata in particolare eseguendo le prove di controllo basato sull'apprendimento per rinforzo, in cui si sono osservate delle oscillazioni periodiche riconducibili proprio ad un ritardo dei dati sensoriali (si veda la Sezione 8.5 per la descrizione del fenomeno).
Nei grafici seguenti è mostrato in ordinata il valore della crosscorrelazione, mentre in ascissa sono riportati i valori dello spostamento temporale per i quali è stato eseguito il calcolo.

I primi test eseguiti valutano l'entità di un eventuale ritardo tra il dato ricavato dai sensori e il dato reale ricavato mediante la tecnica descritta nella sezione precedente. Calcolando il valore della crosscorrelazione su un insieme di campioni raccolti durante il movimento del robot è possibile capire se c'è uno sfasamento tra i due dati: un picco di correlazione in $x = 0$ indica che il legame massimo si ha tra campioni presi nello stesso istante, mentre un picco spostato verso valori di x positivi indica che il legame massimo si ha per un ritardo del dato stimato pari a x.

In Figura 8.3 è riportato il valore della crosscorrelazione tra l'angolo ottenuto tramite il filtro di Kalman e l'inclinazione reale. Il picco in $x = 0$ mostra che non c'è ritardo tra i due dati. La forma della funzione di crosscorrelazione è quella tipica che si ottiene dall'analisi di due segnali identici, confermando la bontà del dato stimato.

In Figura 8.4 sono mostrati i risultati del test effettuato tra la velocità angolare stimata dal filtro di Kalman e quella reale. Il picco della crosscorrelazione tra $x = 0$ e $x = -1$ indicherebbe che ad essere in ritardo è il dato reale. Questo risultato anomalo è dovuto al metodo con cui viene calcolata la velocità angolare reale: l'encoder ottico fornisce una misura dell'angolo, che viene valutata ad ogni ciclo, da cui si ricava la velocità angolare mediante differenza rispetto al valore osservato al ciclo precedente. Questa procedura fa si che all'istante $t = 1$ non si abbia una misura della velocità angolare istantanea, ma si ricavi il valor medio della velocità angolare tra l'istante $t = 0$ e l'istante $t = 1$. Il ritardo del dato reale indicato dall'analisi di crosscorrelazione è quindi corretto, confermando l'ottimo comportamento del filtro di Kalman nello stimare i valori dell'angolo e della velocità angolare.

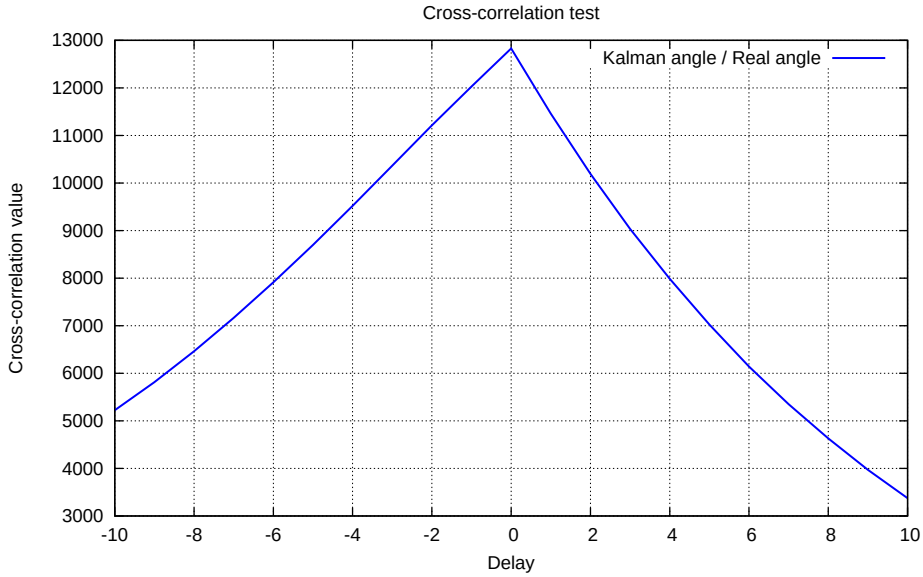


Figura 8.3: Funzione di crosscorrelazione tra angolo stimato dal filtro di Kalman e angolo reale

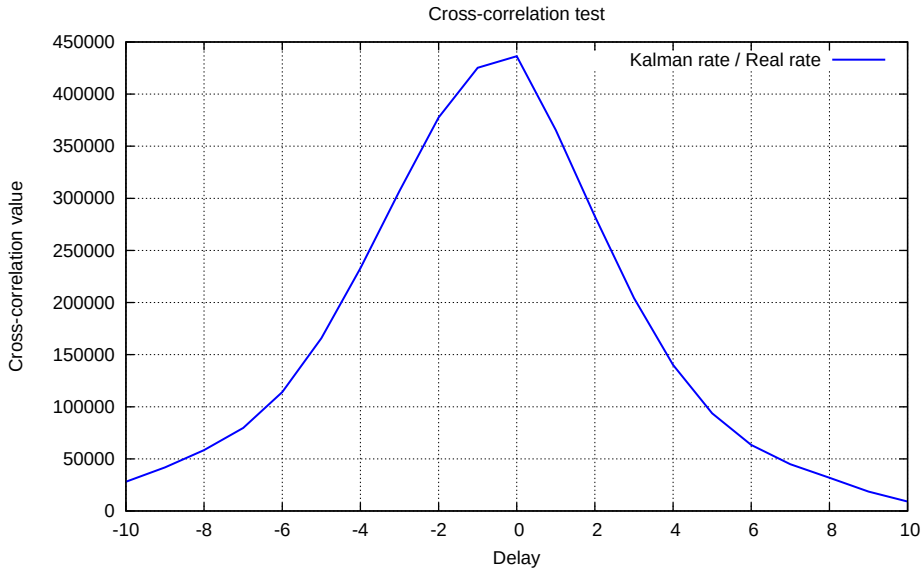


Figura 8.4: Funzione di crosscorrelazione tra velocità angolare stimata dal filtro di Kalman e velocità angolare reale

Il secondo gruppo di prove effettuate è volto a valutare il ritardo tra l'imposizione di un comando ai motori e l'effettivo movimento del robot. Questo ritardo, dovuto alle costanti elettrica e meccanica dei motori e alle caratteristiche meccaniche del sistema, è evidentemente ineliminabile e può portare a problemi nel controllo del robot.
I test sono stati effettuati impartendo comandi casuali ai motori ed eseguendo il calcolo della crosscorrelazione tra i movimenti rilevati e i comandi eseguiti. Il valore di x per cui si osserva il picco della funzione di crosscorrelazione indica il ritardo del movimento del robot rispetto all'istante in cui si è imposto il comando.

In Figura 8.5 è riportata la funzione di crosscorrelazione tra la velocità lineare rilevata dagli encoder ottici utilizzati per l'odometria (si rimanda alla Sezione 5.1.3 per la descrizione del sensore) e il segnale PWM applicato ai motori. Anche in questo caso la velocità di rotazione dei motori è ricavata calcolando la differenza tra il valore incrementale osservato al ciclo corrente e quello osservato al ciclo precedente, che comporta un ritardo intrinseco del dato ottenuto di circa mezzo ciclo.
Il picco di crosscorrelazione si ha per $x = 2$, quindi oltre al ritardo dovuto al metodo utilizzato per il calcolo della velocità si osserva un ulteriore ritardo pari alla durata di un ciclo tra l'istante in cui si imposta il valore del segnale PWM e l'istante in cui l'albero del motore comincia effettivamente a ruotare.

In Figura 8.6 ò mostrato il valore della crosscorrelazione tra la velocità angolare stimata dal filtro di Kalman e il segnale PWM applicato ai motori. Rispetto al test precedente, si aggiunge un altro ciclo di ritardo tra l'istante in cui i motori cominciano a ruotare e l'istante in cui viene rilevato un movimento del telaio. Questo ritardo, imputabile ai giochi della trasmissione, all'elasticità delle ruote e al comportamento dinamico del robot, non può essere eliminato e, se causa di problemi, deve essere compensato dall'algoritmo di controllo.

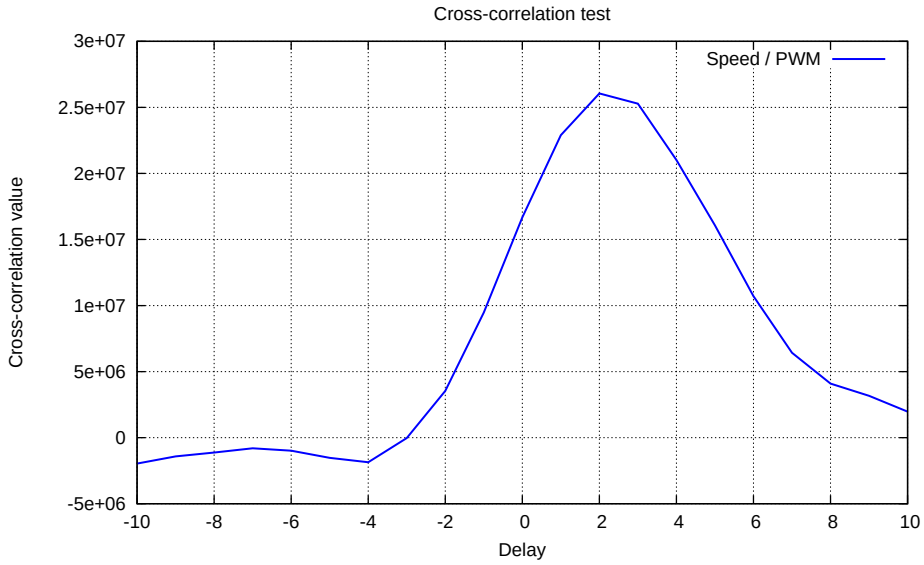


Figura 8.5: Funzione di crosscorrelazione tra velocità lineare e segnale PWM

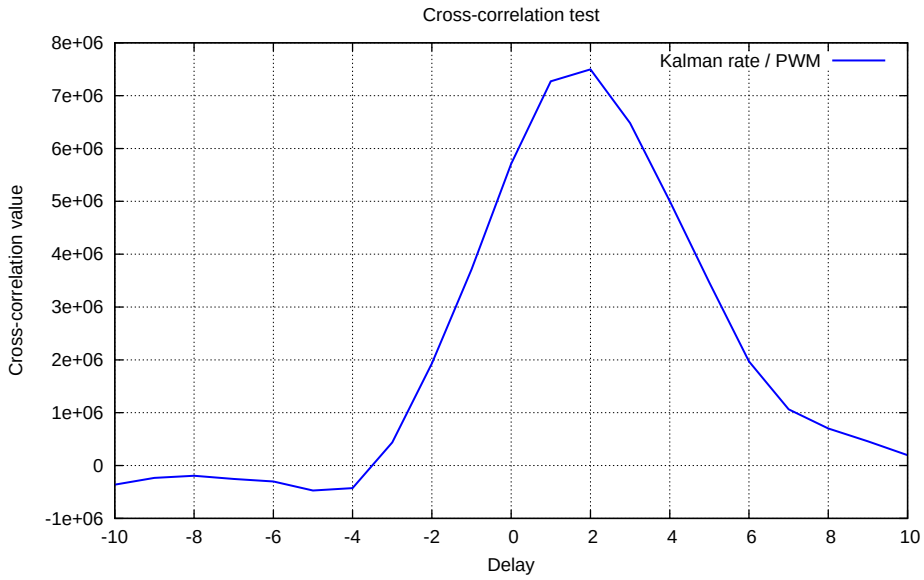


Figura 8.6: Funzione di crosscorrelazione tra velocità angolare e segnale PWM

8.3 Bilanciamento con controllore PID

Si riportano in seguito i risultati ottenuti osservando il comportamento del robot con il controllore PID attivo. Il controllore implementato insegue un setpoint di zero gradi di inclinazione rispetto alla verticale, cercando quindi di mantenere il robot in equilibrio.

In Figura 8.7 sono riportati il valore dell'inclinazione del robot e gli spostamenti effettuati. Si notano piccole oscillazioni, di ampiezza pari a ± 0.3 gradi, corrette provocando degli spostamenti dell'ordine di 2 cm. Osservando il comportamento del robot durante il bilanciamento, si nota come la coppia esercitata dai motori, che è applicata tra telaio e ruote, nel caso di oscillazioni contenute permetta di riportare il telaio nella posizione corretta senza praticamente far girare le ruote.

In Figura 8.8 è mostrato il comportamento del robot in seguito ad una sollecitazione dovuta ad una spinta. Nell'istante $t = 2$ viene esercitata sulla parte superiore del telaio una forza ad esso perpendicolare, che comporta un'inclinazione iniziale pari a 1 grado. Il controllore recupera la condizione di equilibrio in circa 4 secondi, a seguito di uno spostamento del robot di circa 70 cm.

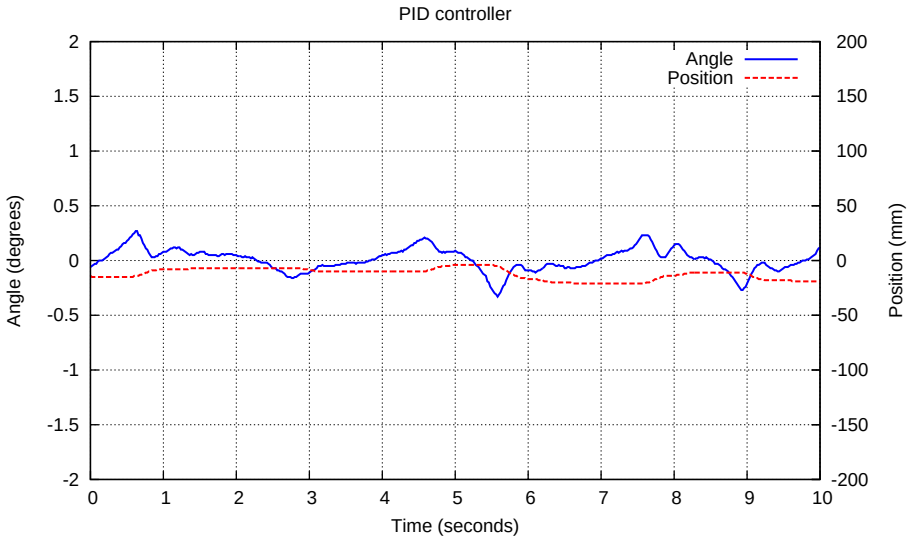


Figura 8.7: Controllore PID: mantenimento dell'equilibrio

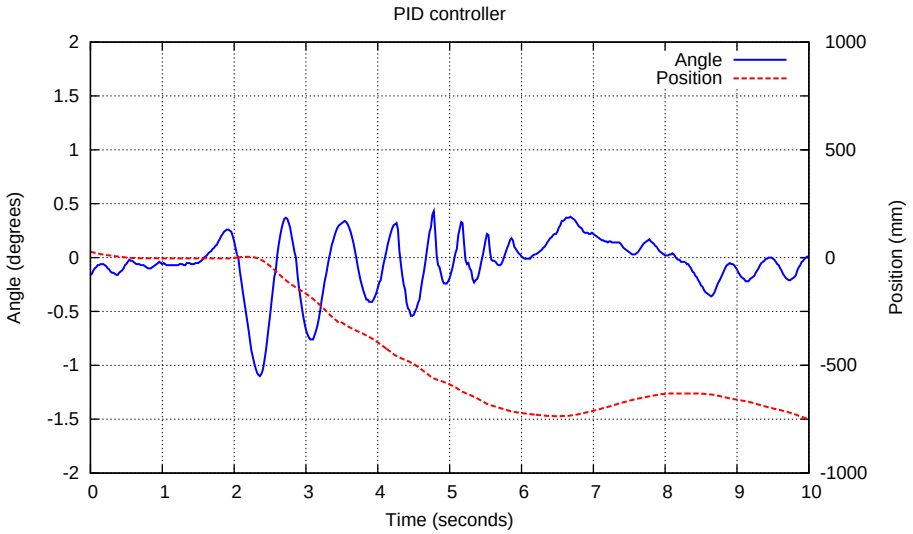


Figura 8.8: Controllore PID: reazione ad una sollecitazione esterna

8.4 Bilanciamento con controllore LQR

Il controllore LQR implementato permette di inseguire un profilo di velocità dato. In seguito sono riportati i grafici ottenuti imponendo un gradino di velocità del valore di 0.4 m/s e della durata di 5 secondi, applicato all'istante $t = 10$.

In figura 8.9 sono mostrati i valori di inclinazione e velocità del robot. Nell'istante $t = 10$, in cui viene modificato il setpoint di velocità, si nota come il robot acceleri innanzitutto nella direzione opposta, facendo si che il telaio si inclini dalla parte corretta permettendo il moto nella direzione richiesta. Successivamente l'inclinazione viene mantenuta costante e il robot si muove con una velocità di circa 0.3-0.4 m/s per 5 secondi. Quando il setpoint varia nuovamente (istante $t = 15$) il robot aumenta la propria velocità per portare l'angolo ad un valore negativo che gli permetta di fermarsi. L'ultima parte del grafico mostra il recupero della posizione verticale mediante un'accelerazione nella direzione opposta.

In figura 8.9 sono riportati i valori di inclinazione e posizione del robot relativi alla stessa prova. Anche in questo grafico si può apprezzare il movimento a velocità costante durante tutta la durata del gradino, per poi accelerare nell'istante $t = 15$ permettendo alle ruote di superare il telaio al fine di portare l'angolo ad un valore negativo che permetta al robot di fermarsi.

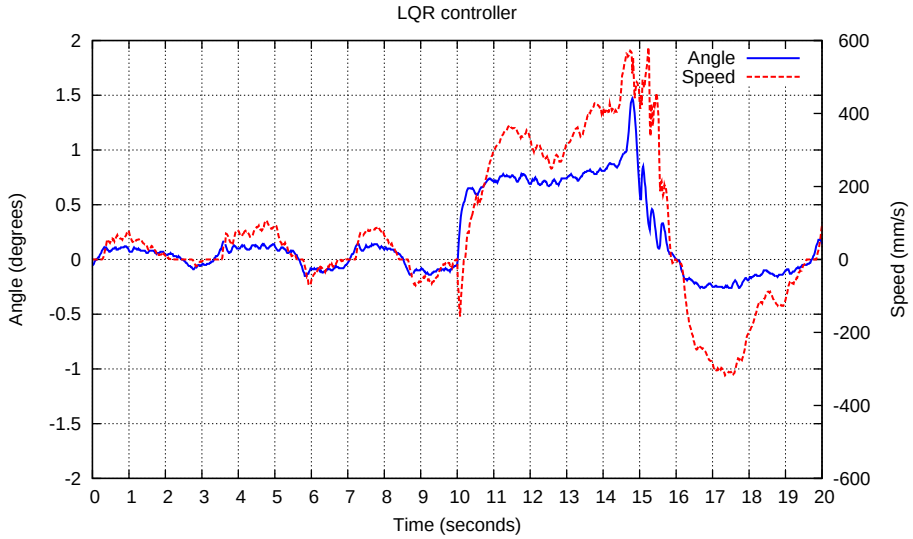


Figura 8.9: Controllore LQR: andamento di angolo e velocità lineare in seguito ad un gradino di velocità

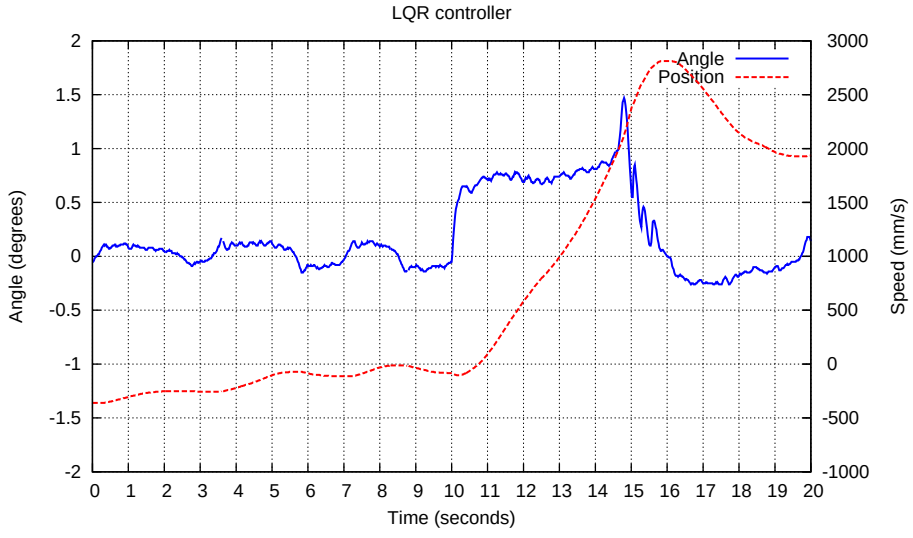


Figura 8.10: Controllore LQR: andamento di angolo e posizione in seguito ad un gradino di velocità

8.5 Bilanciamento con controllore RL

Il controllo del robot tramite un algoritmo basato sulle tecniche di Reinforcement Learning (Sezione 7.3) permette di ricavare una politica di controllo osservando il comportamento del sistema in relazione alle azioni intraprese.

Per permettere all'algoritmo di esplorare lo spazio degli stati e delle azioni sono state registrati i dati forniti dai sensori sottoponendo il robot ad un'azione di controllo completamente casuale. Le azioni, che consistono in un valore PWM casuale e modificato ogni 5 cicli, producono comportamenti differenti che naturalmente si concludono con la caduta del robot. Ripetendo l'operazione più volte sono stati raccolti circa 1000 stati, sulla base dei quali è stato costruito il dataset utilizzato per l'addestramento.
In Figura 8.11 è mostrato l'insieme degli stati che l'algoritmo andrà ad analizzare, ognuno corrispondente ad un determinato valore di angolo, velocità angolare e azione eseguita. La forma della nuvola mostra che lo spazio osservato è abbastanza uniforme.

L'algoritmo procede quindi con l'analisi del cambiamento di stato legato ad ogni azione intrapresa, valutando se lo stato raggiunto comporti un rinforzo positivo, che lo porta più vicino all'obiettivo, oppure una punizione, classificando quell'azione come controproduttiva. Negli esperimenti svolti, che mirano a permettere al robot di apprendere una politica in grado di fargli mantenere l'equilibrio, il rinforzo è definito come:

$$r = \begin{cases} 0 & \text{se} |\alpha| \leq \bar{\alpha} \\ -\alpha & \text{se} |\alpha| > \bar{\alpha} \end{cases}$$

dove $\bar{\alpha}$ è posto pari a 0.2. Questo significa che l'algoritmo viene premiato se l'azione scelta lo porta ad uno stato in cui l'angolo è minore di 0.2 gradi, in valore assoluto, altrimenti la punizione è proporzionale a quanto il nuovo stato è distante dal setpoint.
In Figura 8.12 è mostrata la politica ottenuta in seguito all'apprendimento sul dataset contenente le azioni casuali e i relativi stati osservati. Si nota come, anche con soli 1000 punti a disposizione, l'algoritmo riesca a trovare una politica coerente con l'obiettivo richiesto: in caso di angoli positivi, la scelta è di applicare PWM positivi, che nel sistema di riferimento adottato corrisponde ad una riduzione dell'angolo.

Il procedimento descritto può essere reiterato, al fine di ampliare lo spa-

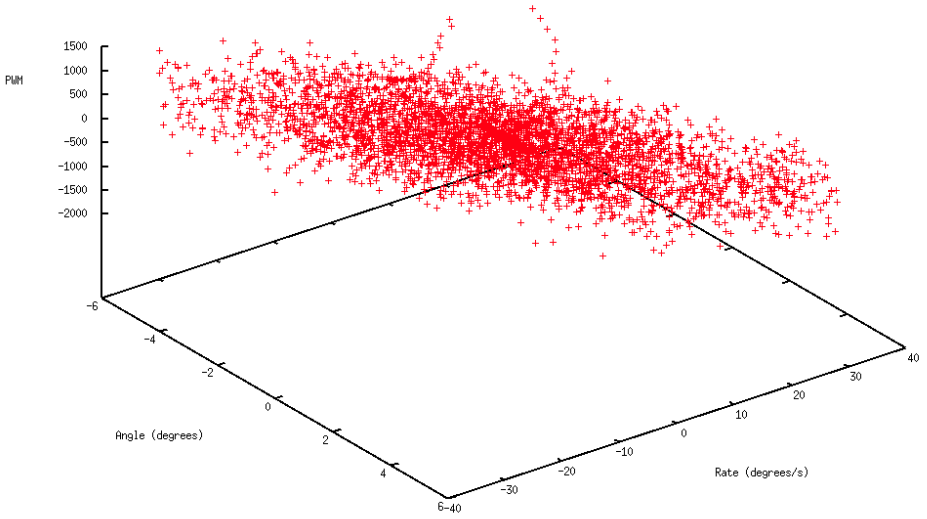


Figura 8.11: Insieme dei punti rappresentati lo spazio visitato durante la fase di esplorazione

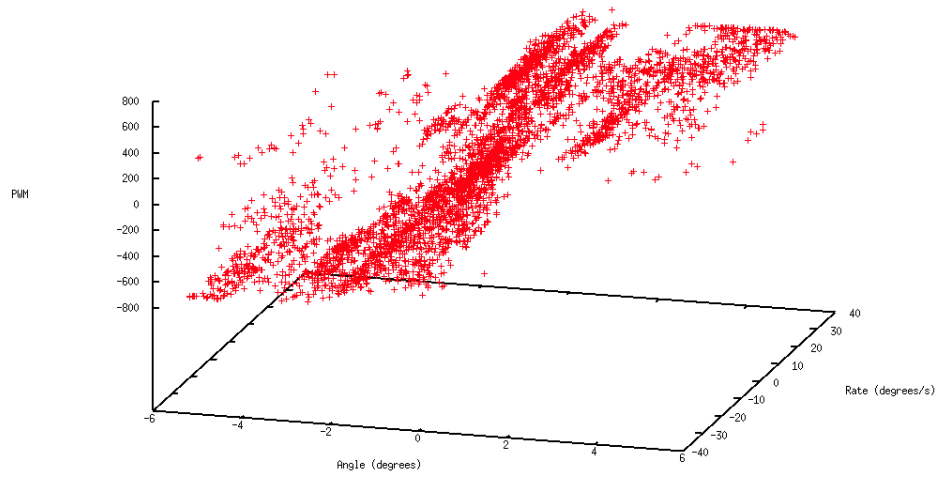


Figura 8.12: Politica di controllo ricavata: azione eseguita in funzione delle variabili di stato

zio degli stati esplorato e di permettere all'algoritmo di raffinare le scelte effettuate, migliorando la politica di controllo ottenuta.
In Figura 8.13 è mostrato il comportamento del robot con il controllore RL attivo, utilizzando la politica ricavata alla prima iterazione, ovvero avendo analizzato i soli stati prodotti dalle azioni casuali. Il robot riesce a non cadere, ma le oscillazioni attorno al punto di equilibrio sono ampie ($\pm 1 - 2$

gradi) e le azioni scelte sono molto brusche, portando il telaio ad inclinarsi dalla parte opposta.
In Figura 8.13 è mostrato il comportamento utilizzando una politica di controllo ricavata dall'analisi di un dataset più vasto. A quello di partenza sono stati infatti aggiunti altri 1000 punti raccolti registrando i dati dei sensori durante l'esecuzione della politica ricavata alla prima iterazione. Si nota come le oscillazioni si sono ridotte, avendo ora ampiezza di $0.5 - 1$ gradi, e le azioni intraprese sono più raramente brusche. Questo miglioramento mostra come l'algoritmo sia in grado di imparare dall'esperienza accumulata, modificando le azioni scelte in caso si siano dimostrate controproducenti. Osservando questi due grafici si possono trarre alcune conclusioni:

- il limite netto tra premio ($|\alpha| \leq 0.2$) e punizione fa si che le azioni siano contenute quando questa condizione è soddisfatta, ma diventino subito più ampie non appena si esce dalla zona premiata
- il comportamento del robot mostra delle oscillazioni a periodo costante

Le oscillazioni sono causate dal ritardo che occorre tra l'applicazione di un'azione e l'osservazione del suo effetto, che l'algoritmo utilizzato non è in grado di interpretare correttamente. Le prime ipotesi per giustificare questo ritardo erano legate ad un ritardo dei dati forniti dai sensori rispetto ai movimenti del robot, ma le analisi di crosscorrelazione riportate nella Sezione 8.2 mostrano che le informazioni rilevate sono corrette. Il ritardo con cui il sistema osserva la variazione di stato a seguito di un'azione, dovuto alle componenti meccaniche, non è quindi eliminabile, ed è necessario modificare l'algoritmo di apprendimento affinchè tenga conto di questa non idealità.

In Figura 8.15 è infine riportato il risultato di una simulazione che sovrappone alla registrazione del comportamento di un controllore PID le azioni che la politica appresa avrebbe effettuato se si fosse trovata negli stessi stati. Le differenze tra l'andamento della variabile di controllo PWM generata dal PID e le azioni scelte dal controllore RL sono le stesse osservate analizzando le prove precedenti: la politica appresa genera delle azioni brusche, che cambiano repentinamente e assumono valori molto elevati non appena si esce dalla zona in cui l'algoritmo viene premiato.

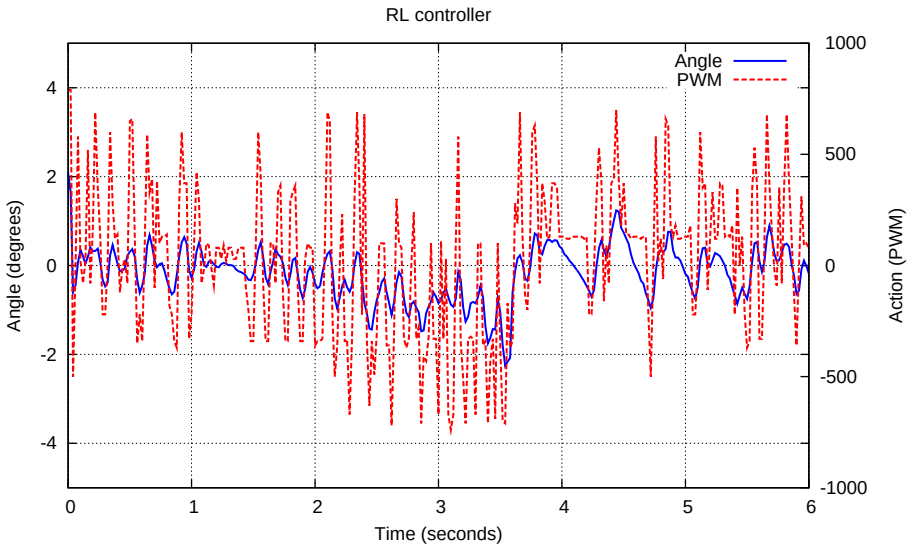


Figura 8.13: Comportamento della politica di controllo alla prima iterazione

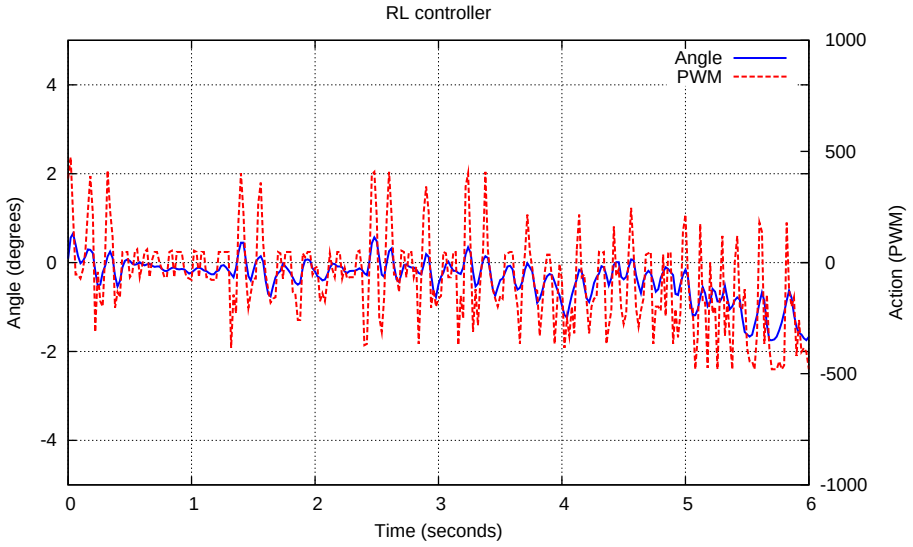


Figura 8.14: Comportamento della politica di controllo alla seconda iterazione

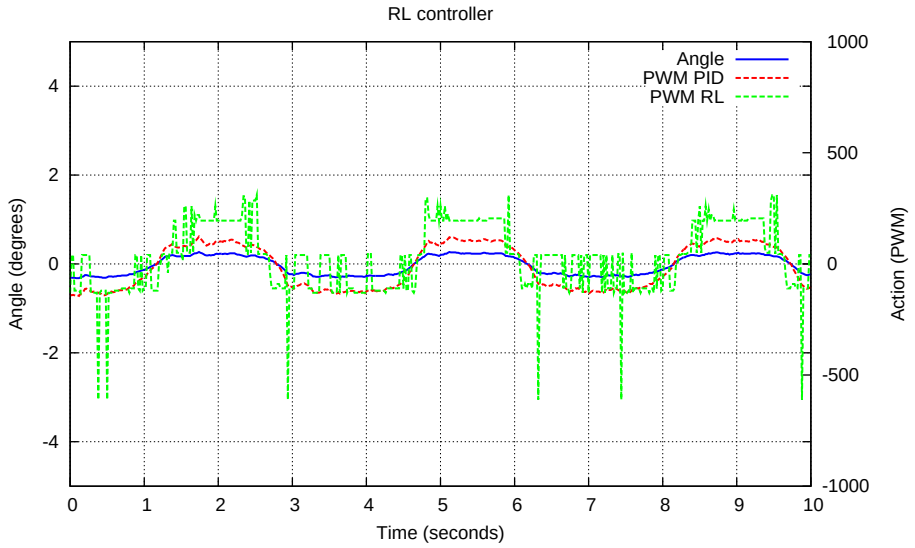


Figura 8.15: Azioni intraprese dalla politica di controllo rispetto al controllore PID

Capitolo 9

Conclusioni e sviluppi futuri

"Tutti sanno che una cosa è impossibile da realizzare, finchè arriva uno sprovveduto che non lo sa e la realizza."

Albert Einstein

9.1 Conclusioni

Il lavoro presentato in questa tesi ha permesso di realizzare una base robotica bilanciante su ruote in grado di mantenere l'equilibrio.
Dal punto di vista meccanico il robot può considerarsi completo, avendo soddisfatto le richieste riguardanti le caratteristiche del telaio, il carico utile trasportabile e la realizzazione di una trasmissione utilizzando componenti il più possibile standard. L'utilizzo di profilati di alluminio modulari permette future modifiche consentendo di montare a bordo eventuali dispositivi accessori.

La scheda elettronica realizzata consente di amplificare e filtrare i dati forniti dai sensori inerziali, ricavando con ottima precisione i valori di angolo e velocità angolare che non presentano ritardi rispetto ai dati reali. I driver di potenza permettono di pilotare i motori elettrici utilizzando diversi tecniche di comando, le cui caratteristiche si adattano ai differenti algoritmi di controllo implementati.

Il software in esecuzione a bordo del robot è scritto per facilitare lo sviluppo di ulteriori funzioni, mettendo a disposizione una serie di strutture di dati relative allo stato e alla configurazione del robot con le quali interagire, dei cicli di controllo eseguiti ad intervallo predefinito e un insieme di funzioni relative alla trasmissione dei dati su porta seriale e all'interpretazione dei comandi ricevuti.

Dal punto di vista del controllo del robot, sono stati studiate e implementate metodologie di tipo classico e tecniche basate sull'apprendimento per rinforzo. Le prove sperimentali effettuate permettono di evidenziare le caratteristiche dei differenti approcci e le problematiche da affrontare al fine di migliorare il comportamento dei diversi algoritmi. I risultati ottenuti mostrano che il robot è in grado di mantenere l'equilibrio sia mediante tecniche di controllo classiche sia apprendendo automaticamente una politica di controllo che gli permetta di non cadere. Il robot è anche in grado di seguire un semplice profilo di velocità, ma i test svolti hanno mostrato che per ottenere risultati soddisfacenti in questa direzione è necessario approfondire lo studio del controllore e realizzare un driver per i motori che consenta un più preciso controllo in coppia.

9.2 Sviluppi futuri

Osservando i risultati delle prove sperimentali è stato possibile identificare le componenti del progetto realizzato che richiedono un ulteriore sviluppo.

Per consentire al robot di inseguire un profilo di velocità complesso, caratteristica fondamentale per il movimento in ambienti vari come quello domestico, è necessario realizzare una scheda di controllo dei motori che permetta un miglior controllo in coppia, rendendo possibile la taratura del controllore LQR in ambiente simulato e la successiva implementazione sul robot reale. È possibile ad esempio implementare un anello di retroazione basato sulla misura della forza elettromotrice generata dal moto del rotore, che in unione al modello del motore permette di risalire al valore della coppia esercitata.

L'algoritmo di controllo basato sull'apprendimento per rinforzo ha mostrato che è possibile far apprendere al sistema quale siano le azioni da eseguire per il mantenimento dell'equilibrio, evidenziando però difficoltà nel produrre una politica di controllo che non produca oscillazioni nel compor-

tamento del robot. Dalle analisi effettuate il problema risulta essere legato al ritardo tra l'applicazione di un'azione e il conseguente movimento del robot, che l'algoritmo non riesce ad interpretare correttamente. È necessario quindi approfondire il processo di apprendimento analizzandone il comportamento al variare del ritardo riscontrato, permettendo di identificare quali modifiche effettuare al fine di migliorarne il comportamento.

Raggiunti risultati soddisfacenti nel controllo del moto è possibile dotare il robot di ulteriori dispositivi che consentano di individuare gli ostacoli, riconoscere gli ambienti, manipolare oggetti e interagire con l'uomo, sfruttando l'agilità e la contenuta impronta a terra che caratterizzano la base bilanciante realizzata.

Bibliografia

[1] T McGeer. Passive dynamic walking. *The International Journal of Robotics Research*, 9(2):62–82, 1990.

[2] Steve Collins, Andy Ruina, Russ Tedrake, and Martijn Wisse. Efficient bipedal robots based on passive-dynamic walkers. *Science Magazine*, 307:1082–1085, 2005.

[3] Anybots. *http://www.anybots.com.*

[4] Dean Kamen. Segway. *http://www.segway.com*, 2001.

[5] D Michie and R. A. Chambers. Boxes: an experiment in adaptive control. *Machine Intelligence*, 2, 1968.

[6] A Barto, R Sutton, and C Anderson. Neuron-like adaptive elements that can solve difficult learning control problems. In *Artificial neural networks: concept learning*, pages 81–93. IEEE Press, Piscataway, NJ, USA, 1983.

[7] Balbots. *http://www.balbots.com.*

[8] David P. Anderson. nbot - a twe wheel balancing robot. *http://www.geology.smu.edu/ dpa-www/robo/nbot/*, 2003.

[9] Ruan Xiaogang and Zhano Jianwei. The pwm servo and lqr control of a dual-wheel upright self-balancing robot. *International Symposiums on Information Processing - Moscow, Russia*, 2008.

[10] Alexander Bogdanov. Optimal control of a double inverted pendulum on a cart. Technical report cse-04-006, Department of Computer Science and Electrical Engineering, Oregon Health and Science University, Oregon, 2004.

[11] Lei Sun Ya and Joo Er Meng. Hybrid fuzzy control of robotics systems. In *Autonomous robotic systems: soft computing and hard computing methodologies and applications*, pages 403–427. Physica-Verlag GmbH, Heidelberg, Germany, Germany, 2003.

[12] Shouju Li, Chenfang Huo, and Yingxi Liu. Inverted pendulum system control by using modified pid neural network. In *ICICIC '08: Proceedings of the 2008 3rd International Conference on Innovative Computing Information and Control*, page 426, Washington, DC, USA, 2008. IEEE Computer Society.

[13] M Riedmiller. Controllong an inverted pendulum by neural plant identification. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 1993.

[14] A Bonarini, C Caccia, A Lazaric, and M Restelli. Batch reinforcement learning for controlling a mobile wheeled pendulum robot. In *Artificial Intelligence in Theory and Practice II*, pages 151–160. Springer Boston, 2008.

[15] Maria Dina Vivarelli. *Appunti di meccanica razionale.* Zanichelli, Milano, 2003.

[16] R. P. Sallen and E. L. Key. A practical method of designing rc active filters. *IRE Transactions on Circuit Theory 2*, pages 74–85, 1955.

[17] ST Microelectronics. Stm32 reference manual. *http://www.st.com/stonline/products/literature/rm/13902.pdf*, 2009.

[18] Paolo Bolzern, Riccardo Scattolini, and Nicola Schiavoni. *Fondamenti di controlli automatici.* McGraw-Hill, Milano, 1998.

[19] *Linear Optimal Control Systems.* Wiley-Interscience, 1972.

[20] A Barto and R Sutton. Time derivative models of pavlovian reinforcement. *Learning and Computational Neuroscience: Foundations of Adaptive Networks*, pages 497–537, 1990.

[21] D. Ormoneit and Ś. Sen. Kernel-based reinforcement learning. *Machine Learning*, 49(2):161–178, 2002.

[22] D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503, 2006.

[23] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63:3–42, 2006.

[24] Leo Breiman. Bagging predictors. *Machine Learning*, 24:123–140, 1996.