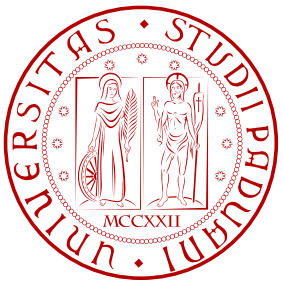
UNIVERSITÀ DEGLI STUDI DI PADOVA



Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria dell'Automazione

Tesi di laurea

Generazione dei riferimenti per algoritmi di motion cueing basati su MPC

Relatore: **Prof. Ing. Alessandro Beghi**
Correlatori: **Ing. Mattia Bruschetta**
Ing. Fabio Maran

Laureando: **Mauro Baseggio**

ANNO ACCADEMICO 2010 – 2011

A mia mamma e mia sorella

Indice

Introduzione

Scopo di un simulatore dinamico di veicolo è quello di riprodurre in modo più fedele possibile gli stimoli di guida che un pilota avrebbe nell'abitacolo reale. Tali stimoli si traducono in termini di percezione di rotazioni e traslazioni che tramite il nostro sistema percettivo collocato sull'orecchio (il *sistema vestibolare*) vengono percepiti principalmente in termini di velocità di rotazione e di accelerazioni traslatorie. Le difficoltà nel riprodurre in maniera appropriata le sensazioni di guida sono essenzialmente legate ai limiti dello spazio di lavoro a disposizione del simulatore stesso, sia in termini di spostamenti longitudinali e laterali, sia in termini di rotazioni lungo i tre gradi di libertà. L'approccio tradizionale è quello illustrato brevemente nella sezione 1.1.2 del primo capitolo adottato dai *washout filters* che consiste essenzialmente nel riprodurre fedelmente le alte frequenze tramite gli spostamenti spaziali della piattaforma mentre le basse frequenze si cerca di riprodurle tramite inclinazioni dell'abitacolo (*tilt coordination*). Tale strategia funziona discretamente bene tuttavia non sfrutta in modo ottimale lo spazio di lavoro e in molti casi l'utilizzo di *roll* e *pitch* (rotazioni attorno agli assi x e y della piattaforma) causa addirittura sensazioni di malessere al pilota (la cosiddetta *motion sickness*) obbligando molte volte a non utilizzare questi due gradi di libertà. La realizzazione quindi di un motion cueing capace di sfruttare appieno le risorse della piattaforma ottimizzando al contempo le sensazioni di guida trasmesse al pilota, è un problema ancora aperto.
L'obiettivo del progetto in cui si inserisce questo lavoro di tesi è quello di realizzare un algoritmo per il motion cueing basato su *MPC* per il simulatore di guida automobilistico in possesso della ditta VI-Grade [18], dalla quale si sono ricevute le specifiche del simulatore stesso e la numerosa mole di dati su cui si sono effettuate le simulazioni e le analisi di questo lavoro di tesi. L'innovazione data dall'utilizzo di questa tecnica di controllo è quella di utilizzare un approccio più metodologico e rigoroso, basato su di un'ottimizzazione vincolata che permette uno sfruttamento massimale della piattaforma secondo un criterio di ottimalità in relazione alla percezione di guida del pilota. Ciò si unisce ai vantaggi propri di un controllo basato su di una predizione, che si adatta particolarmente bene al problema in esame in quanto è possibile cercare di sfruttare la conoscenza che si ha a priori del tracciato o dello stile di guida del pilota.
Questa tesi di laurea, congiuntamente al lavoro svolto parallelamente da Mau-

ro Pozzi [2], si pone quindi come la naturale prosecuzione del lavoro iniziato da Daniele D'Ambrosio [1] in cui si è implementata una versione di MPC che utilizza un ottimizzatore in *c* (il *qpOases*, [19]). Nella tesi del collega Pozzi si indaga in modo approfondito la questione della modellizzazione del sistema vestibolare illustrando in modo dettagliato l'evoluzione dei modelli adoperati negli ultimi decenni, a partire dai primissimi studi fatti in relazione alle problematiche aereonautiche fino ad arrivare a quelli più recenti adattati alle problematiche dei simulatori di guida di veicoli. L'analisi effettuata ha portato ad adoperare il modello proposto da Telban e Cardullo [17]: tale modello, opportunatamente ampliato e adattato come proposto nel lavoro di Chalmers [3] per includere la tilt coordination, costituisce il sistema complessivo su cui fare MPC. Utilizzando poi l'approccio proposto da Renault [13] è stato possibile integrare a tale sistema un primo semplice modello del sistema meccanico, rappresentato dalla piattaforma, in termini di ritardo di attuazione degli ingressi. La linearizzazione del modello vestibolare adoperato e l'aver considerato disaccoppiato l'utilizzo dei sei gradi di libertà del sistema meccanico, ha reso infine possibile suddividire il sistema complessivo in quattro sottosistemi autonomi. Ci si è quindi concentrati nell'approfondire il comportamento dell'accoppiamento *pitch-longitudinale* in quanto rappresenta assieme al sottosistema duale *roll-laterale*, quello di maggior interesse pratico in quanto entrano in azione gli effetti della tilt coordiantion. In questo lavoro di tesi si è quindi analizzato nel dettaglio il comportamento di questo sottosistema, analizzando principalmente il comportamento della piattaforma in termini di sfruttamento dell'area di lavoro in posizione longitudinale e angolare. Si è quindi visto nel dettaglio come i diversi settaggi dati al MPC e le diverse tipologie di generazione dei riferimenti agiscano sulle possibilità del sistema di ottimizzare le risorse della piattaforma. L'analisi effettuata è stata realizzata anche utilizzando una versione leggermente modificata di MPC, capace di decimare l'orizzonte di predizione al fine di eseguire l'ottimizzazione su uno spazio di dimensione inferiore con un notevolissimo risparmio in termini di tempo di calcolo. Inoltre si è proposto un metodo innovativo di generazione online del riferimento appopriato per fare predizione (alternativo a quanto proposto da [3]) in cui si sfrutta in modo intelligente l'informazione a priori di cui si dispone, in termini di caratteristiche del tracciato e stile di guida del pilota, dimostrandone l'efficacia confrontandolo con la predizione ottima data dal segnale stesso.
I risultati complessivamente ottenuti rappresentano un passo importante verso l'implementazione concreta dell'algoritmo di motion cueing proposto nel simulatore reale. Si è cercato infatti di affrontare e di risolvere due delle principali problematiche relativa all'utilizzo del MPC, ovvero l'abbattimento dei tempi di calcolo e la generazione di una riferimento adeguato, elemento fondamentale in un controllo predittivo. Riuscire quindi a disporre di una predizione adeguata, mantenendo al contempo i tempi di calcolo entro i limiti richiesti dal sistema, è uno dei passi fondamentali verso l'utilizzo real time dell'algoritmo di motion cueing proposto.

Capitolo 1

Simulatori dinamici di veicolo

1.1 Strutture esapodali

La piattaforma più utilizzata fin dall'inizio per i primi simulatori di volo è quella di Gough-Stewart. Studi su tale struttura risalgono ai primi anni del 1800 quando il matematico *Cauchy* studiò il moto del cosi detto *ottaedro articolato*. Ci volle più di un secolo, prima con *Gough* e poi con *Stewart*, affinchè le grandi potenzialità di questa struttura potessero essere chiare alla comunità scientifica. La piattaforma di Gough-Stewart (o anche chiamata esapodo per la sua struttura a sei gambe) è un meccanismo parallelo a 6 gradi di libertà (figura 1.1).
Nella sua configurazione elementare è composto da una base mobile collegata al telaio fisso tramite un sistema di attuazione parallelo, in cui gli attuatori lineari, o gambe della piattaforma, sono disposti in modo da avere a due a due un punto in comune alternativamente con la base fissa e quella mobile mediante giunti sferici. I sei arti sono disposti simmetricamente per formare tre quadrilateri con i due lati opposti fissi sulla base mobile e sulla piattaforma fissa, ed i restati due costituiti da due coppie prismatiche. Il successo dell'esapodo, o in generale della meccanica parallela, è dovuto al fatto che a differenza della meccanica seriale fornisce una maggiore capacità di carico di lavoro ripartita tra tutti e 6 gli attuatori, un'ottima accuratezzza nel posizionamento e consente di ottenere elevate accelerazioni per la base mobile.

1.1.1 Primi simulatori

Il primo simulatore di guida automobilistica fu costruito dalla Volkswagen nei primi anni '70, con un sistema a 3 gradi di libertà. Il moto era dato facendo muovere la cabina di pilotaggio lungo i tre assi rotazionali yaw, roll e pitch. Un unico flat screen era montato davanti al pilota seduto nella postazione.
Per vedere, però, il primo simulatore costruito utilizzando un esapodo idraulico bisogna aspettare il 1985 quando Daimler-Benz riusci a costruire una struttura che al tempo fù capace di sviluppare la più realistica senzazione di moto.

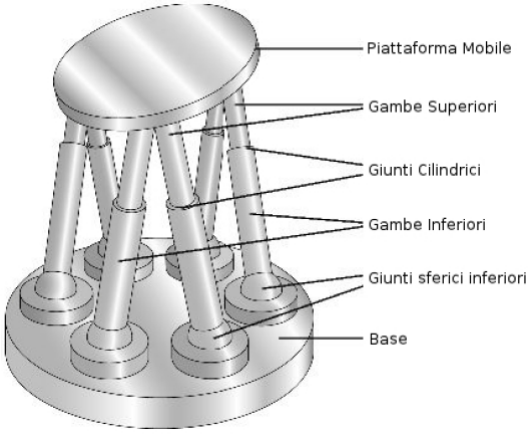


Figura 1.1: Struttura esapodale di Gough-Stewart

La cabina di un auto o un camion erano montate dentro la struttura la quale era fornita di 6 proiettori CRT capaci di dare una visione di 180° al pilota. Nel 1993 il simulatore fu aggiornato in "Advanced Driving Simulator". La differenza principale con il precedente fù la maggiore estensione del moto lungo l'asse y; attraverso un cilindro idraulico si riusciva ad avere un'escursione laterale di 5.6 m.
Dagli inizi degli anni '90 in poi, anche altre case automobilistiche come Toyota, Mazda, BMW, Ford cominciarono ad utilizzare esapodi per i loro simulatori in quanto considerata la soluzione migliore. Inoltre, RENAULT con il simulatore ULTIMATE nel 2004, per incrementare ulteriormente lo spazio a disposizione per le accelerazioni longitudinali, fissa l'esapodo ad un sistema di rotaie capace di muovere l'abitacolo lungo gli assi xy. In questo modo, la piattaforma posizionata in una stanza di 250 m^2 è capace di accelerare fino a ± 7 m/s^2 lungo le rotaie con uno spostamento massimo di 7 m e fino a ± 300 $^\circ/s$ per un massimo di 30 $^\circ$ [4].
Nonostante la dimensione enorme della piattaforma, anche con questo tipo di simulatore non è stato possibile riprodurre con la stessa intensità le accelerazioni, a causa dei vincoli sugli attuatori, che il pilota avrebbe dovuto percepire. Per questo ricerche sempre più avanzate vengono effettuate sul tipo di struttura da utilizzare e per lo sviluppo di nuovi algoritmi di *motion cueing*.

1.1.2 Motion Cueing System

Per far muovere in modo intelligente un simulatore si utilizza un sistema chiamato *motion cueing* o *MCS*, il quale provvede a trasformare i riferimenti di accelerazione $r(t)$ che gli arrivano in ingresso, in possibili segnali da dare agli attuatori. Gli algoritmi di *MCS* hanno dunque degli obiettivi constrastanti da soddisfare al meglio: far percepire al guidatore una sensazione di moto che si avvicini il più possibile alla realtà e mantenere la piattaforma dentro i suoi limiti.
Il primo algoritmo di questo tipo fu realizzato nei primi anni '70 da Schmidt e Con-

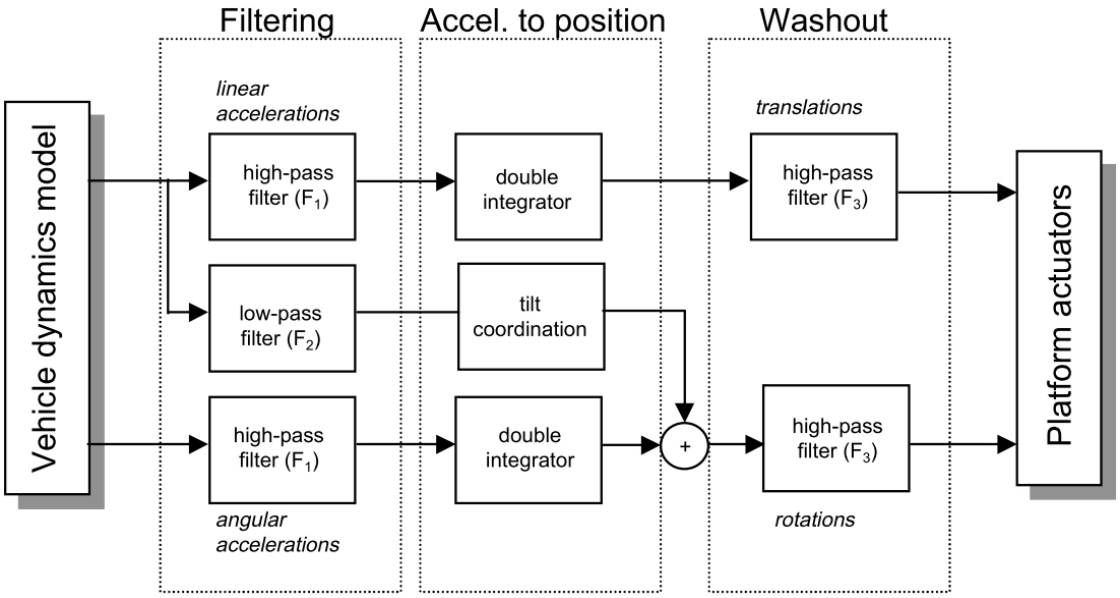


Figura 1.2: Struttura Motion Cueing classico

rad [5] i quali svilupparano la strategia "classica" per *motion cueing* che andava a calcolare le accelerazioni del simulatore lungo tutti i suoi gradi di libertà. Il classico algoritmo di *MCS*, formato da filtri in frequenza e descritto in [7], è strutturato nel seguente modo:

- si rimuovono le basse frequenze dalle accelerazioni longitudinali attraverso un filtro passa-alto, per poi integrare due volte il risultato in modo da avere in uscita il comando di posizione della piattaforma;
- si estraggono dalle accelerazioni le basse frequenze attraverso un filtro passa-basso e da queste si calcola l'angolo di *tilt coordination* che verrà aggiunto al comando di uscita per le posizioni angolari;
- si riporta la piattaforma alla sua posizione neutrale filtrando attraverso un passa-alto le risultanti dei comandi di posizionamento.

Quest'ultimo filtraggio, spesso chiamato anche *motion washout* (Reid and Nahon, 1985) [9], è necessario per evitare che si vada a saturare gli attuatori i quali poi porterebbero a far percepire al pilota false dinamiche del veicolo. Un altro tipo di filtraggio chiamato *anti-backlash filter* può essere aggiunto per ridurre alcune risposte non ottimali in seguito al filtraggio passa-alto (Reymond, 2000) [7].
Utilizzando questo tipo di algoritmo, filtrando le basse frequenze del nostro riferimento le accelerazioni risultanti saranno estremamente ridotte. Questo perchè il settaggio dei parametri del filtro (come guadagno e frequenza di taglio) saranno impostati di sessione in sessione, in funzione del circuito e della guida del pilota,

dato che il *cueing* dovrà fare in modo che non si vada mai a saturare gli attuatori. Qualora, ad esempio, succedesse che l'abitacolo fosse ai margini della posizione possibile, l'algoritmo ha due soluzioni: o restare con la cabina immobile riproducendo solo le accelerazioni angolari, aspettando quindi un'accelerazione in fase per poter tornar indietro alla sua posizione d'origine, oppure andar in controfase facendo percepire al pilota una accelerazione in contrasto a quella ricevuta dal sistema visivo, provocandogli quindi una sensazione di malessere.
In entrambi i casi, le soluzioni adottate non rispettano l'obiettivo dell'algoritmo di moto e quindi si cercherà di tarare il filtro tenendo conto del caso peggiore, facendo in modo che la massima accelerazione in ingresso possa restare entro le soglie degli attuatori della piattaforma.
Come logica conseguenza, in una sessione di guida dove prevalgono nettamente le accelerazioni "normali" verrà utilizzata solo una piccola parte della possibilità di moto del simulatore.
Per ovviare a questo tipo di problema, Parrish e Dieudonne (1975) [6] proposero un metodo *adaptive* poi sviluppato ulteriormente negli anni successivi. Basandosi sul classico *MCS*, invece di usare la strategia solo nel dominio delle frequenze, si considera anche il dominio del tempo: il tipo di filtraggio presente nel precedente algoritmo rimane, ma in questo caso i parametri non sono costati e vengono cambiati ad ogni passo.
Ad ogni istante di campionamento, il guadagno e la frequenza di taglio del classico filtro *MCS* sono derivati dalla minimizzazione di una funzione costo quadratica

$$V_k = (r_k - a_k)^2 + w_1 \cdot v_k^2 + w_2 \cdot p_k^2. \tag{1.1}$$

Il primo termine di V_k è l'errore quadratico tra l'accelerazione del veicolo reale r_k e quella della piattaforma a_k. I restanti due termini sono relativi alla velocità v_k e alla posizione p_k della piattaforma.
Rispetto alla versione precedente, questa *adaptive strategy* risulta sfruttare maggiormente lo spazio di lavoro in condizioni di guida "regolare", ossia con accelerazioni inferiori rispetto i valori con cui vengono tarati i filtri. I pesi w_1 e w_2 definiscono un "trade-off" tra la sensazione di moto che si vuole sviluppare e i limiti imposti dagli attuatori: un aumento di w_1 e w_2 penalizzerà spostamenti e velocità elevate mentre diminuendoli si andrà a favorire la minimizzazione dell'errore quadratico tra le accelerazioni. In ogni caso la taratura andrà effettuata off-line considerando la specifica guida del pilota o del tipo di circuito su cui si andrà ad effetuare la sessione simulativa.
Nel 1982 Sivan e Ish Shalom [10] svilupparono l'algoritmo *ottimo* il quale, in combinazione con il filtraggio lineare passa-basso e passa-alto come per il classico *MCS*, calcola i comandi da dare alla piattaforma minimizzando una funzione costo globale come quella espressa in (1.2):

$$V[u(t)] = \int_0^\infty [(\hat{r}(t) - \hat{a}(t))^2 + w_1 \cdot v^2(t) + w_2 \cdot p^2(t)]dt \tag{1.2}$$

soggetto a

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1.3}$$

con $x = [p\ v\ a\ \hat{a}]^T$

dove (A, B) sono le matrici del modello di percezione del moto, $\hat{a}$ rappresenta l'accelerazione percepita dal pilota all'interno dell'abitacolo e $u(t)$ l'ingresso di controllo del sistema. Per calcolare il minimo della funzione (1.2) occorre conoscere il riferimento lungo un'intera sessione di guida per il tempo $[0, \infty]$.
In questa ultima strategia si ha oltre al modello del sistema considerato, anche il primo utilizzo di una formulazione del modello di percezione del moto proposta negli anni prima da Zacharias (1978); questo perchè dovendo riprodurre la senzasione di moto in una struttura con dei vincoli ben noti, si cerca di andare ad "ingannare" la percezione del pilota in modo da inseguire la traiettoria non facendo uso solo di accelerazioni longitudinali ma anche attraverso l'utilizzo del *tilt-coordination.*

1.2 Desdemona

Parallelamente allo sviluppo di simulatori attraverso la meccanica di Gough-Stewart, sono stati fatti svariati studi per staccarsi dalle classiche strutture con lo scopo di creare una nuova piattaforma più performante, con nuove capacità. Dunque nei primi anni 2000 nasce il progetto *Desdemona.*
Desdemona è un simulatore a base mobile situato presso il TNO (Soesterberg, Olanda), costruito con l'intenzione di dare un voluto disorientamento spaziale, ovvero facendo in modo che il pilota perda il senso "del sopra e del sotto", fondamentale per sessioni simulative sia nel campo aeronautico sia nel campo automobilistico.
Progettato in cooperazione con l'AMST (Ranhofen, Austria) il simulatore si muove lungo i 6 gradi di libertà. La cabina di pilotaggio è montata su una sospensione cardanica, ovvero un insieme di tre anelli, ciascun anello con un paio di cuscinetti, capace di far ruotare la piattaforma attorno a qualsiasi asse rotazionale (φ_c, ψ_c, θ_c, 3 gradi di libertà, $> 2\pi$). Questa sospensione può essere mossa lungo l'asse verticale (H, grado di libertà,$\pm 1m$) e orizzontalmente lungo una rotaia (R, 1 grado di libertà, $\pm 4m$). Tutta la struttura, inoltre, può essere ruotata attorno il proprio asse centrale (ψ_s) in modo da facilitare il moto centrifugo (1 grado di libertà, $< 3g$).
Si riportano, quindi, le specifiche della nuova piattaforma in tabella 1.1.

Tabella 1.1: Desdemona

	ψ_s	R	H	φ_c	ψ_c	θ_c
posizione	illimitata	$\pm 4m$	$\pm 1m$	illimitata	illimitata	illimitata
velocità	$\pm 151°/s$	$\pm 3.2m/s$	$\pm 2m/s$	$\pm 180° \setminus s$	$\pm 180°/s$	$\pm 180°/s$
accelerazione	$\pm 45°/s^2$	$\pm 4.9m/s^2$	$\pm 4.9m/s^2$	$\pm 90°/s^2$	$\pm 90°/s^2$	$\pm 90°/s^2$

Ovviamente per usufruire della performance di questa particolare struttura (raggio

4m) doveva essere realizzato un nuovo *motion cueing*. L'utilizzo di un classico *MCS* o una sua versione estesa non averebbe portato all'utilizzo dell'intero spazio a disposizione; questo perchè quando viene usata una struttura esapodale la cinematica risultante è cartesiana. Per Desdemona, invece, avendone una polare, un apposito *washout filter* è stato progettato chiamato *Spherical Washout Filter* (Wentink, 1985) [8]; caratteristica principale di quest'ultimo è la capacità di sfruttare l'enorme spazio circolare a disposizione di Desdemona, il quale anche in presenza di forti accelerazioni riesce a riprodurle in modo migliore rispetto un esapodo utilizzando sia la *tilt coordination* sia l'accelerazione centrifuga.

1.3 Piattaforma Vi-Grade

Vi-Grade è un'azienda fondata nel 2005, che offre e svilluppa software per simulazioni per tutte le compagnie che vogliono aggiungere un collegamento agli studi ingegneristici teorici e la prova nella realtà di quanto si è studiato.
VI-DriveSim è una nuova linea di prodotti creata da VI-Grade la quale cerca di dare alle compagnie automobilistiche un mezzo di integrazione tra sviluppo di nuovi prototipi o sistemi di controllo e test su una piattaforma che rispecchia l'ambiente reale.
VI-DriveSim si differenzia in due possibili configurazioni: statica e dinamica. Quest'ultima è composta dai seguenti componenti hardware e software:

- piattaforma di moto rivoluzionaria;
- modelli e dati dei veicoli allo stato dell'arte basati sul software già esistente VI-CarRealTime;
- algoritmo di motion cueing;
- provvista di grafica e suoni;
- attuatori che riescono a dare un feedback al controllo pilota.

La piattaforma mobile è basata su una nuova architettura a 6 DOF (*Degree of freedom*) la quale lavora con delle escursioni relativamente ridotte rispetto ai sistemi esapodali descritti precedentemente. In figura 1.3 è rappresentata la piattaforma mobile di VI-DriveSim. In tabella 1.2, inoltre, i valori delle performance del simulatore.
I movimenti longitudinale x e y insieme alla rotazione dello yaw sono disaccoppiati mentre roll, pitch e movimenti lungo l'asse z sono accoppiati tra di loro. VI-DriveSim vuole dunque realizzare un nuovo tipo di approccio per lo sviluppo di una piattaforma di guida, la quale oltre allo scopo principale di test per nuovi sistemi elettronici e training di guida, visto le dimensioni ridotte potrà essere utilizzata per studiare la percezione multisensoriale umana, oppure presso scuole di

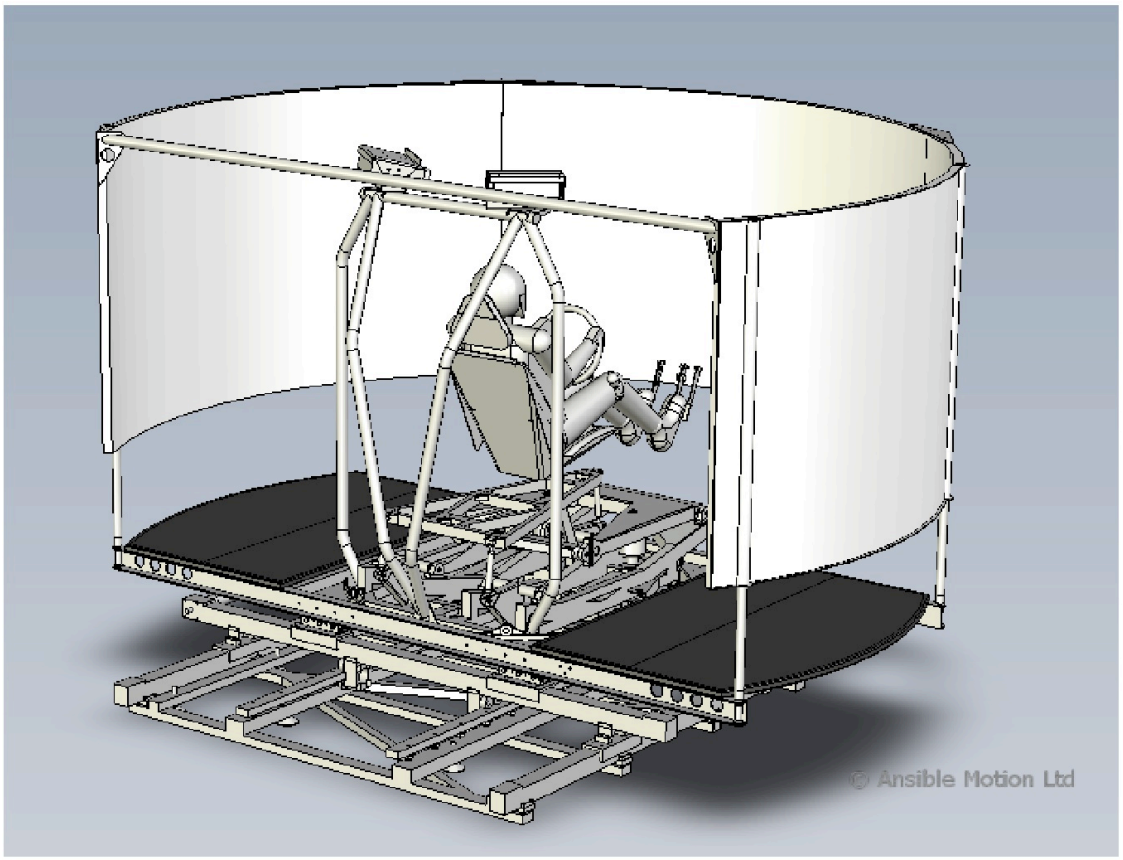


Figura 1.3: Piattaforma VI-Grade

Tabella 1.2: Performance Piattaforma VI-Grade

	Escursione	Velocità	Accelerazione
X	$1m$	$1.3m/s$	$3.3m/s^2$
Y	$1m$	$1.3m/s$	$3.6m/s^2$
Z	$0.3m$	$0.9m/s$	$4.9m/s^2$
Roll	$40\,\mathrm{deg}$	$112\,\mathrm{deg}/s$	$600\,\mathrm{deg}/s^2$
Pitch	$24\,\mathrm{deg}$	$61\,\mathrm{deg}/s$	$600\,\mathrm{deg}/s^2$
Yaw	$40\,\mathrm{deg}$	$61\,\mathrm{deg}/s$	$240\,\mathrm{deg}/s^2$

guida per un primo e sicuro approccio alla guida sulle strade o semplicemente per divertimento.
A causa del poco spazio a disposizone per gli spostamenti, un adeguato *motion cueing* sarà fondamentale per riuscir a riprodurre al meglio le accelerazioni; guardando le escursioni massime elencate nella tabella (1.2), per accelerazioni ad esempio, costanti e a lunga durata, l'utilizzo esclusivo dell'asse x per riprodurre tali valori porterà la cabina in pochi attimi verso i suoi limiti di funzionamento. Per questo motivo si vuole sviluppare un algoritmo che, guardando in avanti di qualche secondo (conoscendo dunque la planimetria della pista), sarà capace di prevedere che non riuscirà a riprodurre fedelmente l'accelerazione utilizzando solamente un asse (x) ma potrà, *ingannando la percezione umana*, anche andare ad agire sul pitch della piattaforma per la riproduzione di segnali a bassa frequenza lasciando che i movimenti di breve durata e improvvisi vengano riprodotti attraverso spostamenti longitudinali.

Capitolo 2

Model Predictive Control

In questo capitolo si introduce il Model Predictive Control (MPC) e se ne illustrano le caratteristiche fondamentali, viene poi illustrato il toolbox esistente in MATLAB e infine si formula il problema in maniera rigorosa e dettagliata. La parte iniziale introduttiva è tratta da [1], [12], [11], [3], per un'analisi più attenta ed esaustiva si suggerisce di leggere con attenzione i libri dedicati all'argomento di Maciejowski [12] e Wang [11].

2.1 Il controllo predittivo

Il controllo predittivo è una strategia nata in ambito industriale sul finire degli anni '70, esso si è subito affermato nell'industria petrolchimica e successivamente è stato studiato attentamente in ambito accademico ed ha trovato applicazione in molteplici settori, grazie alla sua peculiarità di trattare vincoli sul problema da controllare. L'MPC infatti è una tecnica di controllo che fornisce gli ingressi ottimi per un sistema, secondo la logica di minimizzazione di una funzione costo opportuna. Scendendo un pò più nel dettaglio, l'idea che ne sta alla base è quella di calcolare l'azione di controllo (ovvero l'ingresso) risolvendo un problema di controllo ottimo ad anello aperto con orizzonte temporale finito, garantendo il rispetto dei limiti. Tale calcolo viene svolto per ogni iterazione all'interno della finestra temporale fissata. Della sequenza di ingressi ottimi calcolata si applica all'impianto da controllare solo il primo elemento (tecnica *receding horizon*, si veda nel seguito) e si reimposta il problema per il passo successivo, utilizzando il nuovo stato dell'impiato come condizione iniziale.

2.1.1 Vantaggi e svantaggi

Il grande successo del MPC è sicuramente dovuto ai numerosi vantaggi che il suo utilizzo comporta, specialmente in ambito industriale per problemi molto complessi. Uno su tutti, la capacità di gestire vincoli sulle variabili del problema, ovvero

uscite, ingressi e variazioni d'ingresso del sistema. Tale fatto è di importanza cruciale in ambito pratico, dato che spesso si presentano situazioni dove le grandezze in gioco sono limitate (si pensi ad esempio al segnale d'ingresso in corrente di un impianto, oppure alla temperatura massima che non deve essere raggiunta, o ai limiti degli attuatori meccanici...). I vincoli possono inoltre essere distinti in hard e soft, ovvero in vincoli da non violare assolutamente o che invece, magari rimettendoci in termini di costo, si può acconsentire a violare. Il controllo predittivo è altresì uno strumento estremamente versatile: si adatta a problemi SISO come pure a MIMO anche complessi, o a problemi a fase non minima o instabili. Può essere interpretato in modi diversi: per sua natura è un controllo ad anello aperto, ma in certi casi può essere pensato come componente di retroazione; può anche essere utilizzato in ottica di feedforward per compensare i disturbi in ingresso al sistema. Il modo di calcolare l'ingresso ottimo, come già accennato, avviene tramite la scelta di un funzionale di costo, dunque si può scegliere quest'ultimo in base alle esigenze del problema e porre l'accento sulle variabili che necessitano di maggior attenzione. La gestione dei vincoli permette anche, in certi casi, di operare vicino ai vincoli ma in sicurezza, sfruttando così maggiormente le potenzialità dell'impianto.
Il controllo predittivo presenta ovviamente anche alcuni svantaggi, che tuttavia non sono così pesanti da sconsigliarne l'uso. Quello che fino a pochi anni fa era uno svantaggio notevole, ad esempio, lo sta diventando sempre meno: il costo computazionale. Essendo l'MPC un problema di ottimo con vincoli da risolvere ad ogni passo di campionamento dopo aver aggiornato il sistema, in mancanza di strumenti di calcolo adeguati e con problemi onerosi computazionalmente, si capisce come l'MPC presentasse dei limiti intrinsechi nella sua natura. Tuttavia col progresso della tecnologia e con capacità di calcolo sempre maggiori, tale difetto sta diventando sempre meno tale, anche se in alcune applicazioni è ancora presente e il suo utilizzo va trattato con un occhio di riguardo, cosa che è stata fatta ad esempio in questo lavoro di tesi e che verrà discussa più dettagliatamente nel seguito.
Un altro aspetto problematico è la stabilità: essa è di difficile accertamento in termini di un problema con vincoli e dunque non può essere rigorosamente formalizzata, tuttavia finchè è possibile il controllore esercita la sua funzione e dunque si comporta in maniera stabile, anche se una brusca variazione dei valori in gioco potrebbe portare a comportamenti indesiderati. Un ultimo aspetto critico è il fatto che l'MPC usi un modello interno dell'impianto da controllare, su cui fare predizione. Ovviamente tale modello può non essere accurato (nel caso di impianto molto complesso) o subire delle variazioni, dunque l'evoluzione predetta dal controllore potrebbe non essere coerente con quella effettivamente seguita dall'impianto.

2.1.2 L'algoritmo di controllo

Viene ora descritta in modo qualitativo la sequenza dei passaggi necessari per realizzare il controllo predittivo, spiegando in maniera più chiara quanto finora detto.

Modello e predizione

Innanzitutto bisogna stabilire che modello dell'impianto adottare. Inizialmente il controllo predittivo si basava su modelli FIR o di risposta al gradino, che tuttavia consentono di descrivere solamente impianti stabili e spesso risultano di ordine elevato. Quest'ultimo si può ridurre ricorrendo a descrizioni tramite funzioni di trasferimento, che dunque possono essere applicate anche ad impianti instabili, tuttavia sono di difficile gestione nel caso di problemi multivariabili. Questi difetti di tali soluzioni hanno portano negli anni ad affermare l'uso di modelli in spazio di stato, che consentono di sfruttare la teoria dei sistemi lineari e di gestire in maniera efficace sistemi multivariabili. Essi permettono inoltre di inserire in modo semplice la modellizzazione di distrurbi e rumori e di sfruttare la teoria del filtraggio statistico (ovvero un osservatore di stato) nel caso di sistemi con stato non accessibile.
Scelto il modello dell'impianto del problema, si fissano gli orizzonti di predizione H_p e di controllo H_c e si calcola l'evoluzione del sistema in tale finestra temporale, scrivendo le future uscite $y(t+k|t),\ k=1,\dots,H_p$ in funzione degli ingressi futuri $u(t+k|t),\ k=0,\dots,H_c-1$.

Funzionale di costo e calcolo della legge di controllo

L'ingresso ottimo del sistema viene calcolato andando a minimizzare un funzionale di costo, o funzione obiettivo, scelto come detto in precedenza sulla base delle necessità contestuali al problema, con l'obiettivo di fondo di seguire una traiettoria di riferimento, il tutto garantendo il rispetto dei vincoli. Un possibile funzionale di costo è il seguente:

$$J=\sum_{j=1}^{H_p}\delta(j)\left[\hat{y}(t+j|t)-r(t+j)\right]^2+\sum_{j=1}^{H_c}\lambda(j)\left[\Delta u(t+j-1)\right]^2$$

Esso minimizza l'errore $\hat{y}(t+j|t)-r(t+j)$ o la spesa in termini di variazione dell'ingresso $\Delta u(t+j-1)$ a seconda dei valori assunti dai pesi $\delta(j)$ e $\lambda(j)$. Si sottolinea ancora una volta che la scelta del funzionale non è univoca ma arbitraria, l'importante è che sia di tipo quadratico. Data la presenza di vincoli la minimizzazione si traduce in un problema di programmazione quadratica, nel caso eventuale di assenza di vincoli la soluzione si ricava in maniera analitica. Spesso la soluzione del probelma di programmazione quadratica non è così immediata, data la complessità del problema; è dunque prassi consolidata imporre $N_c<N_p$ e assumere che il segnale d'ingresso non subisca più variazioni dopo N_c passi: $\Delta u(t+j-1)=0,\ j\in[N_c,N_p]$. In tal modo la dimensione del problema si riduce con immediata ripercussione sulla sua complessità computazionale.

Applicazione dell'ingresso e tecnica "receding horizon"

Dato che il modello dell'impianto di cui si fa uso potrebbe non essere accurato o che potrebbe esserci presenza di disturbi non misurabili, della sequenza del segnale

di controllo calcolata al paragrafo precedente si applica in realtà solo il primo elemento, $u(t|t)$, mentre il resto è scartato. All'istante iterativo seguente si disporrà del nuovo valore delle uscite e dello stato del sistema sulla base del quale si ripeterà tutto il procedimento fin qui descritto, ottenendo una nuova sequenza di ingressi della quale si applicherà solo il primo elemento, $u(t+1|t+1)$, che in generale sarà diverso da quello calcolato e scartato all'iterazione precedente, $u(t+1|t)$. Tale tecnica prende il nome di receding horizon, dal momento che l'orizzonte di predizione è sempre della stessa durata ma viene "spostato in avanti" di un passo ad ogni iterazione. Si noti che il fatto di avere a disposizione la nuova uscita per effettuare l'ottimizzazione, significa supporre che il modello del sistema in analisi sia strettamente proprio, ovvero che l'uscita $y(k)$ dipenda unicamente dagli ingressi passati e non da quello attuale $u(k)$.

2.2 Model Predictive Control Toolbox 3

In questa sezione viene descritto per sommi capi il toolbox di MATLAB in grado di implementare un controllore MPC. Ciò avviene per spiegare come esista già un potente strumento di controllo in grado di risolvere il problema in questione, ma anche per metterne in evidenza i limiti e le possibilità precluse dal suo utilizzo. Tali fatti giustificheranno la scelta di procedere all'implementazione ed alla realizzazione "in proprio" di un controllore, cosa tra l'altro già messa in evidenza nel lavoro di Daniele d'Ambrosio [1] e quindi qui solamente richiamata. Il contenuto della sezione seguente è tratto in buona parte dalla userguide di Bemporad [14], per completezza il lettore può indagare anche il toolbox precedente, quello di Morari [15], che qui non viene descritto a causa dei limiti che esso presenta.

2.2.1 MPC setup

Il toolbox di Bemporad consente di modellizzare sistemi anche molto complessi. Lo schema generico dell'impianto che si deve controllare è riportato in figura 2.1.
Come si può vedere, il sistema è molto completo e prevede:

- un modello del processo da controllare, in cui gli ingressi sono le "manipulated variables" (MV) ed i disturbi "measured disturbances" (MD) e "unmeasured disturbances" (UD); le uscite si possono dividere in "unmeasured outputs" (UO) e "measured outputs" (MO);
- un modello che genera gli "unmeasured disturbances" di ingresso, a partire da rumore bianco;
- un modello che genera il disturbo di uscita.

Se non si ha bisogno di specificare il modello che genera i disturbi, il controllore assume di default che questi siano generati da integratori pilotati da rumore bianco,

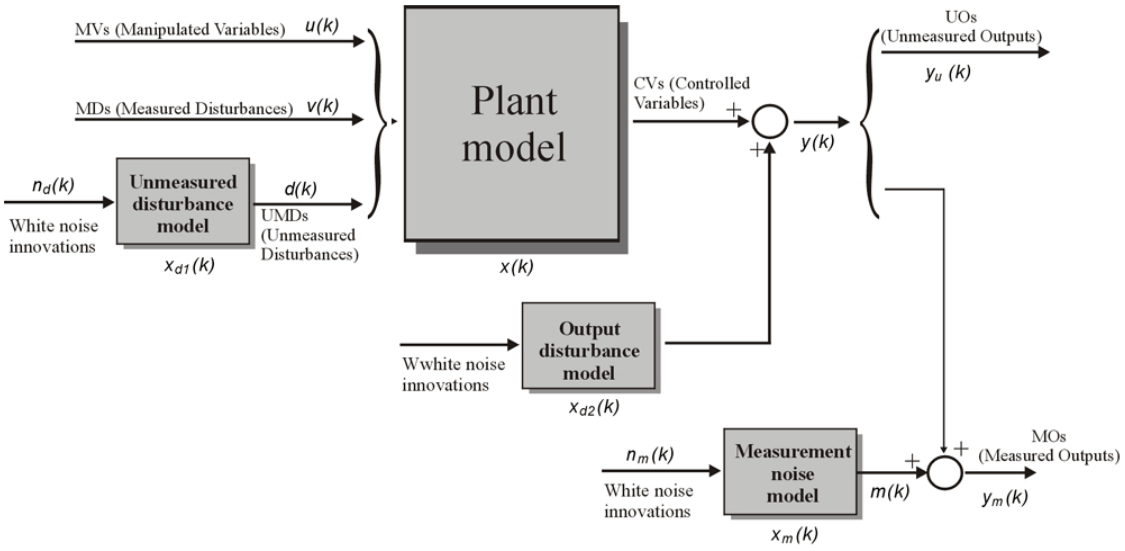


Figura 2.1: Schema generale adottato dal MPC Toolbox

è comunque possibile realizzare la totale assenza di disturbi, sia di ingresso che di uscita.
Il modello del processo è un sistema tempo-invariante lineare descritto dalle seguenti equazioni:

$$\begin{cases} x(k+1) &= A\,x(k) + B_u(k)\,u(k) + B_v(k)\,v(k) + B_d(k)\,d(k) \\ y_m(k) &= C_m\,x(k) + D_{mv}\,v(k) + D_{md}\,d(k) \\ y_u(k) &= C_u\,x(k) + D_{uv}\,v(k) + D_{ud}\,d(k) \end{cases} \tag{2.1}$$

dove $x(k)$ è il vettore n_x-dimensionale dello stato del processo, $u(k)$ è il vettore n_u-dimensionale degli ingressi (MV), $v(k)$ è il vettore n_v-dimensionale dei disturbi misurati (MD), $d(k)$ è il vettore n_d-dimensionale dei disturbi non misurati (UD) che entrano nel processo, $y_m(k)$ è il vettore delle uscite misurate (MO) e y_u è il vettore delle uscite non misurate (UO). La dimensione totale n_y del vettore $y(k)$ è la somma di quelle di $y_m(k)$ e di $y_u(k)$.
L'MPC Toolbox accetta sia processi espressi come sistemi LTI sia modelli ottenuti tramite dati di input/output usando il System Identification Toolbox; unica accortezza da seguire è di usare un modello del processo in cui gli ingressi non abbiano un effetto diretto sulle uscite, come già sottolineato in sezione 2.1.2. Il toolbox prevede anche l'adozione di un modello continuo, che stando alla userguide verrà campionato con lo stesso tempo di campionamento del controllore, tuttavia in fase di test questo fatto ha creato delle difficoltà e si è preferito procedere comunque ad un campionamento preventivo.
Nel caso di modelli con stato non completamente accessibile, il toobox prevede la possibilità di procederne alla stima tramite stimatore lineare con guadagno di Kalman, a partire dalle uscite misurate $y_m(\cdot)$; per fare ciò il toolbox controlla automaticamente che il modello sia (numericamente) osservabile.

Matrici QP

Si ritiene utile riportare la costruzione delle matrici utilizzate nel problema QP perchè il toolbox lavora con un metodo diverso da quello del Wang [11] che si adotta nel presente lavoro di tesi, andando ad effettuare predizione sul modello in spazio di stato di partenza (2.1) e non su quello con riscrittura alternativa dello stato (confrontare i risultati qui ottenuti con quelli di sezione 2.3). Si sottolinea comunque che la tecnica adottata è equivalente, cambia però l'approccio al problema.
Assumiamo per semplicità un modello leggermente più semplice di quello mostrato in precedenza, 2.1:

$$\begin{cases} x(k+1) &= A\,x(k) + B_u(k)\,u(k) + B_v(k)\,v(k) + B_d(k)\,n_d(k) \\ y(k) &= C\,x(k) + D_v\,v(k) + D_d\,n_d(k) \end{cases} \tag{2.2}$$

Al generico istante i l'uscita predetta sarà:

$$y(i|0) = C\left[A^i x(0) + \sum_{h=0}^{i-1} A^{i-1-h}\left(B_u\left(u(-1) + \sum_{j=0}^{h}\Delta u(j)\right) + B_v\,v(h)\right)\right] + B_v\,v(i) \tag{2.3}$$

In forma vettoriale si avrà dunque

$$\begin{bmatrix} y(1) \\ \vdots \\ y(H_p) \end{bmatrix} = S_x\,x(0) + S_{u1}\,u(-1) + S_u \begin{bmatrix} \Delta u(0) \\ \vdots \\ \Delta u(H_p-1) \end{bmatrix} + H_v \begin{bmatrix} v(1) \\ \vdots \\ v(H_p) \end{bmatrix} \tag{2.4}$$

dove la matrice che interessa per il confronto con la 2.14 è S_u:

$$S_u = \begin{bmatrix} CB_u & 0 & 0 & \dots & 0 \\ CB_u + CAB_u & CB_u & 0 & \dots & 0 \\ \vdots & & & & \vdots \\ \sum_{h=0}^{H_p-1} CA^hB_u & \sum_{h=0}^{H_p-2} CA^hB_u & \sum_{h=0}^{H_p-3} CA^hB_u & \dots & CB_u \end{bmatrix} \tag{2.5}$$

come si può notare, la matrice qui riportata presenta termini in più rispetto alla 2.14 illustrata nel seguito, e tali termini sono dovuti alla scelta di andare in predizione col modello originale, senza riscrittura alternativa.

Ottimizzazione

Il funzionale di costo standard che il toolbox va a minimizzare per trovare l'ingresso ottimo è il seguente:

$$\min_{\Delta u(k|k),\ldots,\Delta u(m-1+k|k),\varepsilon} \left[\sum_{i=0}^{H_p-1} \left(\sum_{j=1}^{n_y} \left| w^y_{i+1,j} \left(y_j(k+i+1|k) - r_j(k+i+1) \right) \right|^2 + \right. \right.$$
$$\left. \left. + \sum_{j=1}^{n_u} \left| w^{\Delta u}_{i,j} \Delta u_j(k+i|k) \right|^2 + \sum_{j=1}^{n_u} \left| w^u_{i,j} \big(u_j(k+i|k) - u_{j_{target}}(k+i) \big) \right|^2 \right) + \rho_\varepsilon \varepsilon^2 \right] \tag{2.6}$$

Esso è soggetto ai vincoli

$$\begin{gathered} u_{j_{min}}(i) - \varepsilon V^u_{j_{min}}(i) \leq u_j(k+i|k) \leq u_{j_{max}}(i) + \varepsilon V^u_{j_{max}}(i) \\ \Delta u_{j_{min}}(i) - \varepsilon V^{\Delta u}_{j_{min}}(i) \leq \Delta u_j(k+i|k) \leq \Delta u_{j_{max}}(i) + \varepsilon V^{\Delta u}_{j_{max}}(i) \\ y_{j_{min}}(i) - \varepsilon V^y_{j_{min}}(i) \leq y_j(k+i+1|k) \leq y_{j_{max}}(i) + \varepsilon V^y_{j_{max}}(i) \\ \Delta u(k+h|k) = 0, \quad h = m, \ldots, p-1 \\ \varepsilon \geq 0 \end{gathered} \tag{2.7}$$

$w^{\Delta u}_{i,j}$, $w^u_{i,j}$, $w^y_{i,j}$ sono i pesi non negativi rispettivamente per le variabili Δu, u e y; più il peso è piccolo meno importanza si dà al termine corrispondente nella funzione obiettivo durante la minimizzazione; si noti come i pesi possano essere tempo varianti. Nell'equazione 2.6 i vincoli su u, Δu e y sono resi soft grazie all'introduzione della variabile di slack ε che, con il suo peso ρ_ε, consente di sforare i vincoli: se il peso è infinito essi sono hard, altrimenti possono essere violati dall'ottimizzatore, proporzionalmente al valore di ε, come risulta dalla scrittura dei limiti 2.7. Grazie a tale fatto il problema QP sarà quasi sempre ammissibile, se accadesse che per qualche ragione numerica non lo fosse, il toolbox utilizzerebbe come ingresso il secondo elemento della sequenza ottima calcolata all'istante precedente. Di default il toolbox prevede che i vincoli in uscita siano soft e quelli in ingresso hard.
Per quanto riguarda il riferimento, quando esso non è noto in avanti, si utilizza il valore attuale $r(k)$ per tutto l'orizzonte di predizione, ovvero $r(k+i+1) = r(k)$. Nel MPC la conoscenza della traiettoria futura è chiamata anche *look-ahead* o *anticipative action*. Una simile azione può essere usata anche per i disturbi misurati $v(k)$.
Se il problema non presenta vincoli, la soluzione è calcolata analiticamente, altrimenti il toolbox sfrutta il risolutore di QP di MATLAB qpdantz.

Caratteristiche d'utilizzo

Per poter essere utilizzato, il toolbox prevede sia un comodo ambiente grafico sia di essere gestito tramite istruzioni da riga di comando che in molti casi possono

risultare più comode e versatili da gestire. In tal caso però tutte le grandezze in gioco devono essere inserite in un oggetto di tipo *MPC*, basato su architettura a celle. Ciò non risulta sempre così immediato ed è necessaria una discreta dose di pratica prima di tarare il controllore come desiderato, specialmente per quanto riguarda la gestione di vincoli hard e soft. Altra osservazione va fatta sul toolbox in sè: esso, per quanto concepito in modo abbastanza generico, è pur sempre una scatola chiusa ed eventuali accorgimenti non previsti non possono essere relizzati, ad esempio la presenza di vincoli sullo stato, come spiegato in [1], non è fattibile col toolbox ma lo risulta col controllore sviluppato in questo lavoro di tesi. Dunque la concezione del toolbox come scatola chiusa, la sua proprietà (di copyright) e l'esigenza di risolutori più efficienti in termini di tempo di calcolo ha spinto alla realizzazione di un controllore sviluppato in proprio, nato in ambiente MATLAB e confrontato costantemente col toolbox di Bemporad per avere certezza di comportamento corretto, almeno fintanto che le performance richieste potessero essere raggiunte da entrambi.

2.3 Formulazione del problema MPC

Si procede ora all'implementazione formale e rigorosa del controllo MPC per un modello in spazio di stato, partendo da quanto proposto da Wang ([11]) e realizzato da Daniele d'Ambrosio ([1]) ma integrando con parti in più per pesare tutte le variabili in gioco. Tale procedimento è quello seguito nell'implementazione del controllore oggetto della presente tesi. Sia dunque dato un sistema MIMO discreto del tipo

$$\begin{cases} x_m(k+1) = A_m\, x(k) + B_m\, u(k) \\ y(k) = C_m\, x(k) \end{cases} \tag{2.8}$$

dove il vettore degli ingressi u ha dimensione n_{in}, quello delle uscite y ha dimensione n_{out} e x_m è la variabile di stato assunta di dimensione n; si è implicitamente assunto che l'ingresso non influenzi direttamente le variabili di uscita. Come spiegato da [12], è utile al fine dell'implementazione avere in ingresso la variazione di stato $\Delta u(k)$, a tal scopo si adotta la seconda tecnica proposta sempre in [12] per riscrivere il modello 2.8 in forma alternativa. Si prende dunque la differenza dello stato tra istanti successivi

$$\begin{aligned} \Delta x_m(k+1) &= x_m(k+1) - x_m(k) \\ &= A_m\Big(x_m(k) - x_m(k-1)\Big) + B_m\Big(u(k) - u(k-1)\Big) \end{aligned} \tag{2.9}$$

e definendo la variazione d'ingresso

$$\Delta u(k) = u(k) - u(k-1)$$

si riscrive la 2.9 come

$$\Delta x_m(k+1) = A_m\, \Delta x_m(k) + B_m\, \Delta u(k)$$

A questo punto il modello non ha più in ingresso $u(k)$ ma $\Delta u(k)$; il prossimo passo è quello di collegare $\Delta x_m(k)$ all'uscita $y(k)$; si definisce dunque la nuova variabile di stato

$$x(k) = \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix}$$

Inoltre considerando

$$\begin{aligned} y(k+1) - y(k) &= C_m\Big(x_m(k+1) - x_m(k)\Big) = C_m \Delta x_m(k+1) \\ &= C_m\, A_m\, \Delta x_m(k) + C_m\, B_m \Delta u(k) \end{aligned}$$

si ottiene il seguente modello aumentato:

$$\begin{aligned} \begin{bmatrix} \Delta x_m(k+1) \\ y(k+1) \end{bmatrix} &= \overbrace{\begin{bmatrix} A_m & 0_m^T \\ C_m\, A_m & I \end{bmatrix}}^{A} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} + \overbrace{\begin{bmatrix} B_m \\ C_m\, B_m \end{bmatrix}}^{B} \Delta u(k) \\ y(k) &= \underbrace{\begin{bmatrix} 0_m & I \end{bmatrix}}_{C} \begin{bmatrix} \Delta x_m(k) \\ y(k) \end{bmatrix} \end{aligned} \tag{2.10}$$

con $0_m \in \mathbb{R}^{n_{in} \times n}$ e I matrice identità di dimensione pari a quella dello stato.
A questo punto, detto N_p l'orizzonte di predizione e N_c quello di controllo, con $N_c \leq N_p$, si va in predizione calcolando le uscite future in funzione della traiettoria di controllo $\Delta u(k)$, $\Delta u(k+1)$ ... $\Delta u(k+N_c-1)$, dove k rappresenta l'istante attuale di campionamento. In primo luogo si ha che l'evoluzione di stato del sistema 2.10 è data da:

$$\begin{aligned} x(k+1|k) &= A\, x(k) + B\Delta u(k) \\ x(k+2|k) &= A\, x(k+1|k) + B\Delta u(k+1) = \\ &= A^2\, x(k) + AB\Delta u(k) + B\Delta u(k+1) \\ &\vdots \\ x(k+N_p|k) &= A^{N_p}\, x(k) + A^{N_p-1} B\Delta u(k) + A^{N_p-2} B\Delta u(k+1) + \ldots + \\ &\quad + A^{N_p-N_c} B\Delta u(k+N_c-1) \end{aligned} \tag{2.11}$$

di conseguenza le future variabili d'uscita sono date da:

$$\begin{aligned} y(k+1|k) &= CA\, x(k) + CB\Delta u(k) \\ y(k+2|k) &= CA\, x(k+1|k) + CB\Delta u(k+1) = \\ &= CA^2\, x(k) + CAB\Delta u(k) + CB\Delta u(k+1) \\ &\vdots \\ y(k+N_p|k) &= CA^{N_p}\, x(k) + CA^{N_p-1} B\Delta u(k) + CA^{N_p-2} B\Delta u(k+1) + \ldots + \\ &\quad + CA^{N_p-N_c} B\Delta u(k+N_c-1) \end{aligned} \tag{2.12}$$

Si noti che tutte le uscite future sono scritte in funzione dello stato attuale e dei futuri ingressi $\Delta u(k+i)$, $i = 0, \ldots, N_c - 1$. Esse possono essere scritte in forma vettoriale come

$$Y = Fx(k) + \Phi \Delta U \tag{2.13}$$

dove

$$F = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^{N_p} \end{bmatrix}, \Phi = \begin{bmatrix} CB & 0 & 0 & \ldots & 0 \\ CAB & CB & 0 & \ldots & 0 \\ CA^2B & CAB & CB & \ldots & 0 \\ \vdots & & & & \vdots \\ CA^{N_p-1}B & CA^{N_p-2}B & CA^{N_p-3}B & \ldots & CA^{N_p-N_c}B \end{bmatrix} \tag{2.14}$$

avendo definito i vettori

$$Y = \begin{bmatrix} y(k+1|k) \\ y(k+2|k) \\ \vdots \\ y(k+N_p|k) \end{bmatrix} \in \mathbb{R}^{(N_p \cdot n_{out}) \times 1} \qquad \Delta U = \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix} \in \mathbb{R}^{(N_c \cdot n_{in}) \times 1} \tag{2.15}$$

Definito il modello in spazio di stato e dato un riferimento $r(k)$, l'obiettivo del controllo predittivo è portare il sistema, ovvero l'uscita, il più possibile vicino al riferimento assegnato, andando a minimizzare un funzionale di costo dipendente dalla sequenza di ingressi incogniti ΔU. Il funzionale adottato è il seguente:

$$J(\Delta U) = (R_s - Y)^T Q (R_s - Y) + \Delta U^T R \, \Delta U + U^T S \, U \tag{2.16}$$

Il primo termine serve per minimizzare l'errore tra le uscite predette ed il riferimento, mentre i rimanenti pesano gli ingressi e le loro variazioni. Le matrici Q, R e S sono rispettivamente le matrici contenenti i pesi sull'errore, sulla variazione dell'ingresso e sull'ingresso complessivo, esse hanno dimensioni opportune dipendenti dal numero di ingressi e uscite e dai passi di predizione e controllo, possono essere "statiche", ovvero composte da pesi sempre uguali, oppure "dinamiche", nel senso che i pesi variano con gli istanti di predizione. Considerazioni in merito alla scelta di tali matrici sono rimandate alle sezioni RIF!!! La matrice R_s è un vettore di dimensioni pari a quelle di Y e contiene il riferimento; se non si dispone di informazioni future su di esso essa contiene solamente il valore attuale $r(k)$ supposto costante per tutta la durata della predizione, se invece si fa utilizzo di una strategia look-ahead allora essa riporta l'evoluzione del riferimento $r(k+1)$, $r(k+2)$, $\ldots$ $r(k+N_p)$.
Per risolvere il problema innanzitutto è necessario riscrivere il funzionale di costo 2.16 in modo che dipenda unicamente da ΔU. Il vettore degli ingressi U può essere

riscritto nella seguente forma:

$$\begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \\ \vdots \\ u(k+N_c-1) \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & \dots & 0 \\ I & I & 0 & \dots & 0 \\ I & I & I & \dots & 0 \\ \vdots & & & & \vdots \\ I & I & I & \dots & I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \Delta u(k+2) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix} + \begin{bmatrix} u(k-1) \\ u(k-1) \\ u(k-1) \\ \vdots \\ u(k-1) \end{bmatrix} \tag{2.17}$$

o in forma vettoriale come

$$U = T\Delta U + U_i \tag{2.18}$$

Sostituendo la 2.18 e la 2.13 nella 2.16, si riscrive il funzionale nella maniera desiderata:

$$J = (R_s - Fx - \Phi\Delta U)^T Q(R_s - Fx - \Phi\Delta U) + \Delta U^T R\,\Delta U + (T\Delta U + U_i)^T S\,(T\Delta U + U_i) \tag{2.19}$$

Dopo un pò di passaggi algebrici, la 2.19 si riduce nella seguente forma, nella quale sono stati trascurati i termini non dipendenti da ΔU, dato che i funzionali possono essere definiti a meno di una costante:

$$J = \Delta U^T \left(\Phi^T Q\,\Phi + R + T^T S\,T\right)\Delta U + 2\Delta U^T \left(\Phi^T Q(R_s - Fx(k)) + T^T S\,U_i\right) \tag{2.20}$$

Il funzionale è ora espresso nella forma classica per un problema di programmazione quadratica (QP):

$$J = \frac{1}{2}\,\Delta U^T H \Delta U + \Delta U^T F \tag{2.21}$$

con

$$H = 2\left(\Phi^T Q\,\Phi + R + T^T S\,T\right) \quad F = 2\left(\Phi^T Q(R_s - Fx(k)) + T^T S\,U_i\right) \tag{2.22}$$

2.3.1 Caso non vincolato

Si risolve ora il problema in questione nel caso, estremamente semplice, di assenza di vincoli. Sotto tale ipotesi non è necessario procedere alla risoluzione di un problema QP dal momento che la soluzione si ricava in maniera analitica, infatti il valore che minimizza il funzionale 2.21 si ottiene derivando il funzionale stesso rispetto a ΔU e ponendo la derivata pari a zero:

$$\frac{\partial J}{\partial \Delta U} = 0 \quad \Rightarrow \quad \Delta U_{ott} = H^{-1}F \tag{2.23}$$

Si è fatta l'assunzione implicita che l'inversa della matrice H esista. La soluzione trovata in 2.23 contiene tutta la legge ottima di controllo, ma in base al principio del receding horizon ad ogni singolo passo di campionamento si calcola un set d'ingresso ΔU_{ott} per i prossimi N_c passi, del quale si utilizza soltanto il primo elemento:

$$\Delta u_{ott}(k) = [I_{n_{in}}\ \ 0_{n_{in}}\ \dots\ 0_{n_{in}}]\,\Delta U_{ott}$$

In seguito il minimo viene ricalcolato considerando i cambiamenti avuti nel sistema al passo successivo, tutto ciò deve avvenire in tempo reale. Per interpretare il risultato 2.23 in termini di controllo a retroazione, aspetto che ai fini della presente tesi non è molto rilevante, si veda [11].

2.3.2 Vincoli in ingresso e uscita

La situazione più interessante del controllo predittivo è ovviamente quella, più complessa, di presenza di limiti su ingressi e uscite. In tal caso al problema QP 2.21 si aggiunge una disuguaglianza del tipo

$$A\,\Delta U \leq b \tag{2.24}$$

con A e b matrici opportune contenenti tutte le informazioni sui vincoli. La 2.24 è giustamente scritta rispetto alla variabile ΔU, su cui si opera la minimizzazione del funzionale 2.21. È dunque necessario procedere alla traduzione di tutti i vincoli sul sistema per riferirli alla variazione d'ingresso, prima di dare in pasto il problema QP ad un ottimizzatore.

Vincoli sulla variazione di ingresso

È questo il caso più semplice che si presenta, in quanto non necessita di "conversione" dei vincoli visto che riguardano già ΔU. Si suppone, qui e nei casi seguenti, che i vincoli agiscano in termini di doppia disuguaglianza del tipo $\Delta u_{min} \leq \Delta u(\cdot) \leq \Delta u_{max}$ (con Δu_{min} e Δu_{max} vettori di dimensione pari a quella dell'ingresso); la strategia che si adotta è quella di spezzare quest'ultima in due disuguaglianze nello stesso verso, del tipo 2.24, in modo tale da ricondursi poi a quest'ultima scrittura. In questo caso si pone dunque

$$\begin{bmatrix} I & 0 & \dots & 0 \\ 0 & I & \dots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \dots & I \\ -I & 0 & \dots & 0 \\ 0 & -I & \dots & 0 \\ \vdots & & & \vdots \\ 0 & 0 & \dots & -I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix} \leq \begin{bmatrix} \Delta u^{max} \\ \Delta u^{max} \\ \vdots \\ \Delta u^{max} \\ -\Delta u^{min} \\ -\Delta u^{min} \\ \vdots \\ -\Delta u^{min} \end{bmatrix} \tag{2.25}$$

La matrice identità che compare ha dimensioni opportune; la disuguaglianza appena scritta viene indicata in forma vettoriale come $M_1\,\Delta U \leq N_1$.

Vincoli sull'ingresso

Per quanto riguarda i vincoli in ingresso, anch'essi nella forma di doppia disuguaglianza, ovvero $L_1 \leq u(\cdot) \leq L_2$, si ricorre alla scrittura 2.17 già usata in prece-

denza per tradurre il vincolo sulla variazione d'ingresso. Procedendo come nel caso precedente si ottiene

$$\begin{bmatrix} I & 0 & \dots & 0 \\ I & I & \dots & 0 \\ \vdots & & & \vdots \\ I & I & \dots & I \\ -I & 0 & \dots & 0 \\ -I & -I & \dots & 0 \\ \vdots & & & \vdots \\ -I & -I & \dots & -I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+N_c-1) \end{bmatrix} \leq \begin{bmatrix} L_2 - u(k-1) \\ L_2 - u(k-1) \\ \vdots \\ L_2 - u(k-1) \\ -L_1 + u(k-1) \\ -L_1 + u(k-1) \\ \vdots \\ -L_1 + u(k-1) \end{bmatrix} \tag{2.26}$$

In analogia col caso precedente, si battezzano le due matrici qui introdotte come M_2 e N_2. Si noti che N_2, a differenza delle altre fin qui introdotte, dipende dalla condizione iniziale $u(k-1)$, con k istante attuale di calcolo. Dunque tale matrice non può essere definita offline come le altre sin qui introdotte ma deve essere ridefinita ad ogni passo, dal momento che l'ingresso evolve diventando ad ogni iterazione la condizione iniziale dell'istante successivo.

Vincoli sull'uscita

In questo caso, volendo sfruttare la relazione 2.13, si adotta una notazione leggermente diversa rispetto ai due casi precedenti. Supponiamo infatti che il vettore delle uscite abbia dei limiti inferiori e superiori Y_{min} e Y_{max} rispettivamente, ovvero che valga la relazione $Y_{min} \leq Y \leq Y_{max}$, con i vettori dei limiti non più di dimensione pari a quella dell'uscita ma a quella del prodotto di quest'ultima per l'orizzonte di predizione (come indicato nella 2.15). A questo punto si può sfruttare la 2.13 e riscrivere in modo immediato i limiti in funzione della variazione d'ingresso:

$$\begin{bmatrix} \Phi \\ -\Phi \end{bmatrix} \Delta U \leq \begin{bmatrix} Y^{max} - Fx(k) \\ -Y^{min} + Fx(k) \end{bmatrix} \tag{2.27}$$

Le matrici introdotte vengono identificate come M_3 e N_3, per N_3 valgono i discorsi fatti nel caso precedente in merito ad N_2, con la differenza che ora è lo stato $x(k)$ che varia ad ogni iterazione.

Programmazione Quadratica

Una volta tradotti i vincoli come richiesto, si può passare alla risoluzione del problema tramite la programmazione quadratica. Per chiarezza si riportano qui le formule già scritte in precedenza: si vuole trovare l'ingresso ottimo che minimizzi il funzionale

$$J = \frac{1}{2} \Delta U^T H \Delta U + \Delta U^T F \tag{2.28}$$

soggetto al vincolo di disuguaglianza

$$A\,\Delta U \leq b \qquad \text{con} \qquad A = \begin{bmatrix} M_1 \\ M_2 \\ M_3 \end{bmatrix} \qquad b = \begin{bmatrix} N_1 \\ N_2 \\ N_3 \end{bmatrix} \tag{2.29}$$

nel quale sono state costruite le matrici A e b imponendo i vincoli del problema in questione. A questo punto il problema di programmazione quadratica viene risolto automaticamente da un ottimizzatore e dunque in questo lavoro di tesi non si entra nei dettagli specifici; qualora il lettore desideri farlo può rifarsi a [1] o a [12].
Nell'implementazione eseguita durante il lavoro di tesi, ci si è avvalsi dell'ottimizzatore qpOASES reperito sul web [19], dal momento che i tradizionali ottimizzatori di MATLAB risultano più lenti ed il problema del calcolo in tempo reale era particolarmente sentito, data la necessità di implementare il controllore sul simulatore VI-grade.

Capitolo 3

Realizzazione del modello per l'MPC

Una delle problematiche chiave per l'implementazione del controllo predittivo è la scelta del modello su cui fare predizione. E' fondamentale disporre di un modello adeguato al problema in analisi per ottenere dei risultati significativi. Nel lavoro in questione ci si è soffermati sulla modellizzazione del sistema meccanico rappresentato dalla piattaforma mobile e sulla scelta del modello del sistema percettivo, descritto ampiamente nella tesi del collega Mauro Pozzi [2]. In questo capitolo si introduce dunque il sistema meccanico e si crea il modello complessivo che si andrà ad utilizzare, facendo opportune considerazioni ed arrivando ad un modello disaccoppiato che faciliti il lavoro e migliori le prestazioni.

3.1 Sintesi del sistema serie meccanico-vestibolare

Utilizzando l'approccio proposto dall'articolo di Renault [13], si è deciso di utilizzare come modello complessivo su cui fare MPC il sistema ottenuto facendo la serie del modello del sistema meccanico della piattaforma e del modello del sistema vestibolare. Gli ingressi del sistema meccanico sono le accelerazioni longitudinali ed angolari applicate alla piattaforma che rappresentano l'ingresso esogeno con cui si controlla la dinamica complessiva, mentre le uscite (che sono gli ingressi del modello vestibolare) sono le accelerazioni longitudinali e le velocità angolari che, dopo il "filtro" del sistema meccanico, arrivano alla persona seduta nell'abitacolo, la quale le percepisce attraverso il proprio sistema vestibolare. Le uscite del sistema complessivo sono quindi le accelerazioni longitudinali e le velocità angolari percepite dal pilota. L'obiettivo del motion cueing, implementato tramite MPC, sarà quello di determinare gli ingressi esogeni che minimizzano l'errore tra accelerazioni e velocità effettivamente percepite e quelle ottenute filtrando attraverso il sistema vestibolare il riferimento di telemetria adottato.

3.1.1 Modello del sistema meccanico

Non disponendo di un modello adeguato della piattaforma si è utilizzato lo stesso approccio del lavoro di tesi di Daniele D'ambrosio [1]: ci si è limitati a considerare il ritardo di attuazione del sistema meccanico attraverso un semplice modello continuo del secondo ordine:

$$\ddot{y} + 2\xi\omega_0\dot{y} + \omega_0^2 y = \omega_0^2 u \tag{3.1}$$

con u che rappresenta l'accelerazione (longitudinale o angolare) applicata alla piattaforma lungo un grado di libertà (x, y o z) e y l'accelerazione effettivamente riprodotta. La funzione di trasferimento della 3.1 è la seguente:

$$W(s) = \frac{\omega_0^2}{s^2 + 2\xi\omega_0 s + \omega_0^2}$$

che realizzata in forma canonica di controllo produce le seguenti matrici:

$$A_{a,\omega_i} = \begin{bmatrix} 0 & 1 \\ -\omega_0^2 & -2\xi\omega_0 \end{bmatrix} \qquad B_{a,\omega_i} = \begin{bmatrix} 0 \\ \omega_0^2 \end{bmatrix} \qquad C_{a,\omega_i} = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D_{a,\omega_i} = 0 \tag{3.2}$$

Ognuno dei sei gradi di libertà (a_x, a_y, a_z, $\dot{\omega}_x$, $\dot{\omega}_y$, $\dot{\omega}_z$) con cui si può comandare la piattaforma viene dunque filtrato col modello appena ricavato, con l'accortezza però che per quanto riguarda la dinamica angolare l'ingresso è in accelerazione ($\dot{\omega}_i$) ma l'uscita desiderata è in velocità (ω_i), pertanto il modello va modificato come segue:

$$A_{\omega_i} = \begin{bmatrix} 0 & 1 & 0 \\ -\omega_0^2 & -2\xi\omega_0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \qquad B_{\omega_i} = \begin{bmatrix} 0 \\ \omega_0^2 \\ 0 \end{bmatrix} \qquad C_{\omega_i} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \quad D_{\omega_i} = 0 \tag{3.3}$$

Le dinamiche 3.2 e 3.3 introdotte possono essere considerate una buona approssimazione dell'attuazione meccanica della piattaforma, come già affermato all'inizio di questa sezione. Tuttavia avendo a che fare con un problema di dimensioni notevoli, con molte variabili in gioco (pesi, limiti) e non facilmente intuibile in alcuni suoi comportamenti (per quanto riguarda l'MPC) si è deciso, nelle fasi di lavoro di tesi, di eliminare la dinamica introdotta e modellare il sistema meccanico come un semplice ritardo (discreto) di attuazione pari ad un passo, superando così l'inconveniente ulteriore di dover discretizzare i modelli 3.2 e 3.3, fatto che introdurrebbe ulteriore complicazione nella comprensione delle dinamiche del problema. In tal modo è risultato infatti più semplice analizzare e comprendere i meccanismi del MPC ed i comportamenti del sistema complessivo. Le matrici con cui si modella effettivamente il sistema meccanico per un singolo grado di libertà sono dunque le seguenti:

$$A_{a,\omega_i} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B_{a,\omega_i} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad C_{a,\omega_i} = \begin{bmatrix} 0 & 1 \end{bmatrix} \quad D_{a,\omega_i} = 0 \tag{3.4}$$

Non vengono più apportate distinzioni per gli ingressi angolari e dunque questi ultimi saranno in velocità e non più in accelerazione. Come sviluppo futuro non è da escludere una modellizzazione accurata del sistema meccanico vero (o l'utilizzo di una funzione del secondo ordine stile 3.1) e la sua introduzione nel problema per affinare l'algoritmo di cueing, ma in questa fase, piuttosto complessa, dell'introduzione del modello vestibolare e della comprensione dei meccanismi del MPC si è come detto preferito farne a meno. Il modello complessivo a sei gradi di libertà del sistema meccanico, visto come semplice ritardo, sarà quindi:

$$\Sigma_m = (A_m, B_m, C_m, D_m) \tag{3.5}$$

con:

$$A_m = \begin{bmatrix} A_{a_x} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & A_{a_y} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & A_{a_z} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & A_{\omega_x} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & A_{\omega_y} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & 0_{2\times2} & A_{\omega_z} \end{bmatrix} \quad B_m = \begin{bmatrix} B_{a_x} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & B_{a_y} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & 0_{2\times1} & B_{a_z} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & B_{\omega_x} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & B_{\omega_y} & 0_{2\times1} \\ 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & 0_{2\times1} & B_{\omega_z} \end{bmatrix} \tag{3.6}$$

$$C_m = \begin{bmatrix} C_{a_x} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & C_{a_y} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & C_{a_z} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & C_{\omega_x} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & C_{\omega_y} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & 0_{1\times2} & C_{\omega_z} \end{bmatrix} \quad D_m = 0_{6\times6} \tag{3.7}$$

con $U_m^T = [\, a_x \;\; a_y \;\; a_z \;\; \omega_x \;\; \omega_y \;\; \omega_z \,]$ e $Y_m^T = [\, a_x \;\; a_y \;\; a_z \;\; \omega_x \;\; \omega_y \;\; \omega_z \,]$, dove con U_m si intendono propriamente gli ingressi esogeni in accelerazione e velocità angolare del sistema, mentre con Y_m si intendono gli stessi parametri filtrati dal sistema meccanico descritto e effettivamente applicati alla piattaforma.

3.1.2 Modello del sistema percettivo

Con riferimento alla tesi gemella di Mauro P. [2] il modello del sistema vestibolare adottato è quello di Telban e Cardullo [16]. Le funzioni di trasferimento ottenute, rispettivamente per le velocità angolari e le accelerazioni longitudinali, sono:

$$W_{s,i}(s) = \frac{\hat{\omega_i}(s)}{\omega_i(s)} = \frac{s^2}{s^2 + s\left(\frac{1}{T_a} + \frac{1}{T_L}\right) + \frac{1}{T_a T_L}} \tag{3.8}$$

$$W_{o,i}(s) = \frac{\hat{f}_i(s)}{f_i(s)} = \frac{K\tau_a}{\tau_l \tau_s} \frac{s + \frac{1}{\tau_a}}{s^2 + s\left(\frac{1}{\tau_s} + \frac{1}{\tau_L}\right) + \frac{1}{\tau_s \tau_L}} \tag{3.9}$$

dove con $\hat{\omega_i}$, $\hat{f}_i$ si intendono rispettivamente la velocità angolare e l'accelerazione longitudinale percepite lungo l'i-esimo grado di libertà ($i = x, y, z$).
Dovendo inserire il sistema vestibolare all'interno dell'algoritmo del MPC, si rende

necessario fornirne una realizzazione in spazio di stato. Per i canali semicircolari la realizzazione, ottenuta coi comandi MATLAB, porta alle seguenti matrici:

$$A_{s,i} = \begin{bmatrix} -\frac{1}{T_a} - \frac{1}{T_L} & 1 \\ -\frac{1}{T_a T_L} & 0 \end{bmatrix} \quad B_{s,i} = \begin{bmatrix} -\frac{1}{T_a} - \frac{1}{T_L} \\ \frac{1}{T_a T_L} \end{bmatrix} \quad C_{s,i} = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D_{s,i} = \begin{bmatrix} 1 \end{bmatrix} \tag{3.10}$$

Per le macule[1], essendo la funzione di trasferimento strettamente propria, si procede ad una realizzazione in forma canonica d'osservabilità:

$$A_{o,i} = \begin{bmatrix} -\frac{1}{\tau_s} - \frac{1}{\tau_L} & 1 \\ -\frac{1}{\tau_s \tau_L} & 0 \end{bmatrix} \quad B_{o,i} = \begin{bmatrix} \frac{K\tau_a}{\tau_l \tau_s} \\ \frac{K}{\tau_l \tau_s} \end{bmatrix} \quad C_{o,i} = \begin{bmatrix} 1 & 0 \end{bmatrix} \quad D_{o,i} = \begin{bmatrix} 0 \end{bmatrix} \tag{3.11}$$

Come detto precedentente, i modelli 3.10 e 3.11 riportati si riferiscono ad un singolo asse ma il problema in analisi si sviluppa in tutte e tre le dimensioni. È dunque necessario procedere all'accoppiamento degli assi per poter descrivere il modello complessivo del sistema vestibolare.
Per i canali semicircolari si ha:

$$A_s = \begin{bmatrix} A_{s,x} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & A_{s,y} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & A_{s,z} \end{bmatrix} \quad B_s = \begin{bmatrix} B_{s,x} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & B_{s,y} & 0_{2\times2} \\ 0_{2\times1} & 0_{2\times1} & B_{s,z} \end{bmatrix} \quad C_s = \begin{bmatrix} C_{s,x} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & C_{s,y} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & C_{s,z} \end{bmatrix}$$

e

$$D_s = \begin{bmatrix} D_{s,x} & 0 & 0 \\ 0 & D_{s,y} & 0 \\ 0 & 0 & D_{s,z} \end{bmatrix} \quad x_s = \begin{bmatrix} x_{s,x} \\ x_{s,y} \\ x_{s,z} \end{bmatrix} \quad y_s = \begin{bmatrix} \hat{\omega}_x \\ \hat{\omega}_y \\ \hat{\omega}_z \end{bmatrix} \quad u_s = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{3.12}$$

Seguendo l'approccio proposto da Chalmers [3], in ipotesi di angoli piccoli, è possibile approssimare le velocità angolari ω_i con le velocità $\dot{\beta}_i$:

$$\omega_i = \dot{\beta}_i \tag{3.13}$$

dove con β_i si intendono gli angoli di Eulero che mappano il sistema di riferimento solidale alla piattaforma rispetto a quello assoluto inerziale (rispettivamente *roll*, *pitch*, *yaw* per x, y, z). Tale approssimazione risulta adeguata nel problema in questione, dove le variazioni angolari permesse alla piattaforma non superano gli $0.25rad$. In questo modo il modello complessivo finale per i canalisemicirolari diventa:

$$\begin{cases} x_s(t+1) = A_s x(t) + B_s \dot{\beta}(t) \\ \hat{w} = C_s x(t) + D_s \dot{\beta}(t) \end{cases} \tag{3.14}$$

[1]In inglese, le macule si indicano con il termine "otoliths", che è errato tradurre con "otoliti" in quanto tale termine, in italiano, indica i cristalli presenti sopra la membrana otolitica, chiamati in inglese "otoconials". Tuttavia, per continuità con la letteratura, si è deciso di adottare i pedici "o" e "oto" per identificare le matrici relative al modello delle macule.

Per quanto riguarda le macule, con procedimento analogo si ottiene che il modello complessivo è dato da:

$$A_{o_f} = \begin{bmatrix} A_{o,x} & 0_{2\times2} & 0_{2\times2} \\ 0_{2\times2} & A_{o,y} & 0_{2\times2} \\ 0_{2\times2} & 0_{2\times2} & A_{o,z} \end{bmatrix} \quad B_{o_f} = \begin{bmatrix} B_{o,x} & 0_{2\times1} & 0_{2\times1} \\ 0_{2\times1} & B_{o,y} & 0_{2\times2} \\ 0_{2\times1} & 0_{2\times1} & B_{o,z} \end{bmatrix} \quad C_{o_f} = \begin{bmatrix} C_{o,x} & 0_{1\times2} & 0_{1\times2} \\ 0_{1\times2} & C_{o,y} & 0_{1\times2} \\ 0_{1\times2} & 0_{1\times2} & C_{o,z} \end{bmatrix}$$

e

$$D_{o_f} = \begin{bmatrix} D_{o,x} & 0 & 0 \\ 0 & D_{o,y} & 0 \\ 0 & 0 & D_{o,z} \end{bmatrix} \quad x_{o_f} = \begin{bmatrix} x_{o,x} \\ x_{o,y} \\ x_{o,z} \end{bmatrix} \quad y_o = \begin{bmatrix} \hat{f}_x \\ \hat{f}_y \\ \hat{f}_z \end{bmatrix} \quad u_o = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} \tag{3.15}$$

Anche in questo caso si rende necessaria una modifica all'ingresso, dal momento che non si dispone delle accelerazioni risultanti f_i ma solo di quelle longitudinali a_i applicate, tuttavia prima di procedere alla modifica delle matrici 3.15 è necessario introdurre il concetto della tilt coordination, elemento fondamentale negli algoritmi di motion cueing per simulatori dinamici di veicolo.

Tilt coordination Come è facile intuire e come Daniele d'Ambrosio ha sperimentato nel suo lavoro di tesi [1], la semplice traslazione lungo gli assi $x - y$ della piattaforma del simulatore non è in grado di riprodurre le accelerazioni in gioco in un problema vero. Tuttavia, grazie al fatto che il corpo umano è incapace di distinguere le accelerazioni traslazionali da quelle gravitazionali con le sole macule (infatti per tale scopo è necessaria un'ulteriore informazione sensoriale, in genere fornita dalla vista) si può ingannare il sistema sensoriale umano andando ad inclinare la piattaforma di un angolo opportuno, dando al pilota la percezione di un'accelerazione che andrà a sommarsi a quella fornita dal moto longitudinale. Tale tecnica è detta tilt coordination.

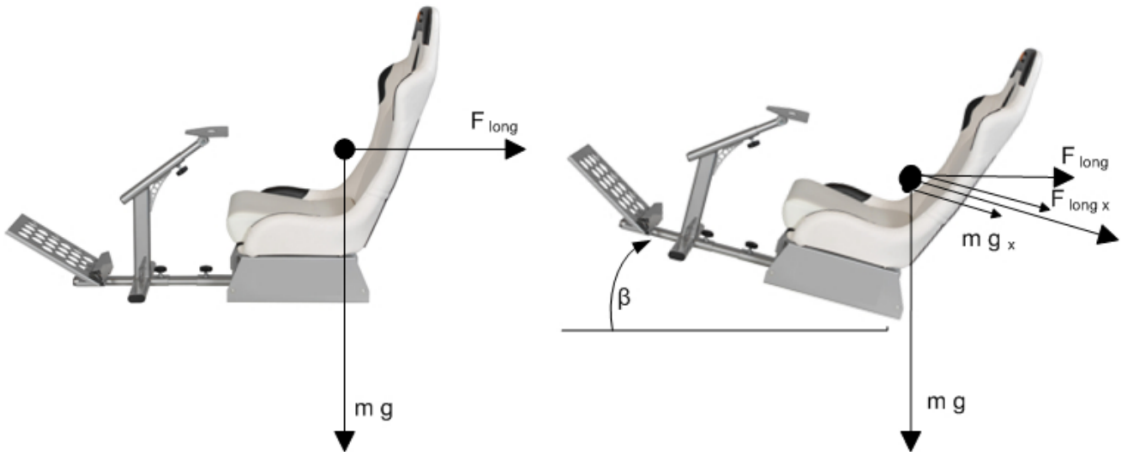


Figura 3.1: Tilt coordination per l'asse x [1].

Come si vede da figura 3.1, inclinando il sedile della piattaforma di un angolo di pitch β, le forze longitudinali e gravitazionali si scompongono e la forza risultante percepita, F_x, risulta essere la somma delle componenti F_{long_x} e mg_x. Più nel dettaglio, assumendo un'inclinazione di un angolo di pitch θ e di uno di roll ϕ (l'angolo di yaw è indifferente per la tilt coordination), si ha che il vettore di accelerazione gravitazionale g, riportato nella terna solidale alla piattaforma del simulatore tramite opportune matrici di rotazione, vale

$$g_s = R_x\, R_y\, g_i = R_x(\beta_x) R_y(\beta_y) \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} -g \sin\theta \\ g \cos\theta \sin\phi \\ g \cos\theta \cos\phi \end{bmatrix} \tag{3.16}$$

dove si sono indicati coi pedici s e i i vettori relativi alla terna del simulatore e inerziale rispettivamente. L'accelerazione complessivamente percepita sarà quindi $f_s = a_s - g_s$:

$$f_s = \begin{bmatrix} f_x \\ f_y \\ f_z \end{bmatrix} = \begin{bmatrix} a_x + g \sin\theta \\ a_y - g \cos\theta \sin\phi \\ a_z - g \cos\theta \cos\phi \end{bmatrix} \approx \begin{bmatrix} a_x + g\theta \\ a_y - g\phi \\ a_z - g \end{bmatrix} \tag{3.17}$$

L'ultima approssimazione è stata fatta linearizzando nell'intorno dello 0 (sotto l'ipotesi quindi di angoli piccoli). La tecnica della tilt coordination, oltre a permettere di realizzare accelerazioni maggiori di quelle simulabili con la sola traslazione, consente se lo si desidera di spezzare il problema in due per risolverlo in maniera più semplificata: si va ad inseguire le alte frequenze del riferimento traslando la piattaforma e le basse inclinandola. Tale soluzione al problema del motion cueing è piuttosto semplice ed elementare e non sfrutta appieno le potenzialità della piattaforma, tuttavia è il principio su cui si basano la maggior parte degli algoritmi di motion cueing presenti in commercio (i cosiddetti *washout filters* di cui si è accennato nel primo capitolo 1).

Inserendo ora nel modello la tilt coordination (utilizzando cioè l'espressione 3.17), è possibile esprimere l'accelerazaione complessiva f_s in funzione delle accelerazioni longitudinali applicate alla piattaforma a_s e degli angoli β_S di inclinazione della stessa rispetto al riferimento assoluto:

$$f_s = H \begin{bmatrix} a \\ \beta \end{bmatrix} \qquad H = \begin{bmatrix} 1 & 0 & 0 & 0 & g & 0 \\ 0 & 1 & 0 & -g & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \tag{3.18}$$

Se gli ingressi desiderati fossero $[a \;\; \beta]^T$, basterebbe a questo punto moltiplicare la matrice B_{o_f} 3.1.2 per la matrice H appena scritta, tuttavia si vuole utilizzare come ingresso la variazione degli angoli di Eulero perchè rappresentano anche l'ingresso per il modello dei canalisemicircolari, ovvero si vuole avere $u = [a \;\; \dot{\beta}]^T$. Tale obiettivo si ottiene aggiungendo allo stato x_{o_f} del sistema tre ulteriori stati, ovvero gli

angoli di Eulero $\beta_x, \beta_y, \beta_z$ e andando ad apportare una leggera modifica alle matrici del sistema. Si partiziona infatti la nuova matrice $B_{o_{a,\beta}}$ nel seguente modo:

$$B_{o_{a,\beta}} = B_{o_f} H = \begin{bmatrix} B_1 & B_2 \end{bmatrix} \tag{3.19}$$

A questo punto è sufficiente includere B_2 nella nuova matrice A_o per ottenere il modello delle macule con in ingresso le variazioni degli angoli di Eulero $\dot{\beta}$:

$$A_o = \left[\begin{array}{ccc|c} A_x & 0_{2\times2} & 0_{2\times2} & \\ 0_{2\times2} & A_y & 0_{2\times2} & B_2 \\ 0_{2\times2} & 0_{2\times2} & A_z & \\ \hline & 0_{3\times6} & & 0_{3\times3} \end{array}\right] \quad x_o = \begin{bmatrix} x_{o_x} \\ x_{o_y} \\ x_{o_z} \\ \beta_x \\ \beta_y \\ \beta_z \end{bmatrix} \quad u_o = \begin{bmatrix} a_x \\ a_y \\ a_z \\ \dot{\beta}_x \\ \dot{\beta}_y \\ \dot{\beta}_z \end{bmatrix} \tag{3.20}$$

e

$$B_o = \begin{bmatrix} B_1 & 0_{6\times3} \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} \quad C_o = \begin{bmatrix} C_{o_f} & 0_{3\times3} \end{bmatrix} \quad D_o = \begin{bmatrix} D_{o_f} & 0_{3\times3} \end{bmatrix} \quad y_o = \begin{bmatrix} \hat{f}_x \\ \hat{f}_y \\ \hat{f}_z \end{bmatrix} \tag{3.21}$$

Modello complessivo

Si può ora procedere alla scrittura del modello vestibolare complessivo, comprendente quello dei canali semicircolari 3.1.2,3.12 e quello delle macule con tilt coordination 3.20, 3.21. Basterebbe includere in un unico modello le matrici ricavate, tuttavia si aggiungono anche sei variabili di stato (posizione e velocità del centro di massa della piattaforma lungo i tre assi x, y, z) per poter in seguito aggiungere i vincoli necessari al MPC. Queste aggiunte sono identificate come "stati integrali" vista la loro evoluzione di stato:

$$\begin{bmatrix} \dot{p_x} \\ v_x \\ p_y \\ v_y \\ p_z \\ v_z \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ v_x \\ p_y \\ v_y \\ p_z \\ v_z \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_x \\ a_y \\ a_z \\ \dot{\beta}_x \\ \dot{\beta}_y \\ \dot{\beta}_z \end{bmatrix} = A_e \begin{bmatrix} p_x \\ v_x \\ p_y \\ v_y \\ p_z \\ v_z \end{bmatrix} + B_e u \tag{3.22}$$

Si aggiunge anche in uscita $\dot{\beta}$, per poterla vincolare nel problema MPC, semplicemente aggingendo una riga nelle matrici di evoluzione dell'uscita. L'evoluzione di stato del sistema vestibolare è dunque così descrivibile:

$$\begin{aligned} \dot{x}_{vest} &= A_{vest}\, x_{vest} + B_{vest}\, u \\ y_{vest} &= C_{vest}\, x_{vest} + D_{vest}\, u \end{aligned} \tag{3.23}$$

con

$$A_{vest} = \begin{bmatrix} A_{scc} & 0_{6\times9} & 0_{6\times6} \\ 0_{9\times6} & A_{oto} & 0_{9\times6} \\ 0_{6\times6} & 0_{6\times9} & A_e \end{bmatrix} \quad B_{vest} = \begin{bmatrix} 0_{6\times3} & B_{scc} \\ B_{oto} & \\ B_e & \end{bmatrix} \tag{3.24}$$

e

$$C_{vest} = \begin{bmatrix} C_{scc} & 0_{3\times9} & 0_{3\times6} \\ 0_{3\times9} & C_{oto} & 0_{3\times6} \\ 0_{9\times9} & 0_{9\times6} & I_{9\times9} \\ & 0_{3\times24} & \end{bmatrix} \quad D_{vest} = \begin{bmatrix} 0_{3\times3} & D_{scc} \\ D_{oto} & \\ 0_{9\times6} & \\ 0_{3\times3} & I_{3\times3} \end{bmatrix} \quad x_{vest} = \begin{bmatrix} x_{scc} \\ x_{oto} \\ p_x \\ v_x \\ p_y \\ v_y \\ p_z \\ v_z \end{bmatrix} \tag{3.25}$$

Le variabili d'ingresso sono $u = [a \;\; \dot{\beta}]^T$, e rappresentano quindi propriamente i sei gradi di libertà di cui dispone il sistema: le tre accelerazioni lungo x, y, z, e le tre rotazioni lungo gli stessi assi (che, come detto precedentemente sono dette *roll*, *pitch*, *yaw*); mentre l'uscita è il vettore seguente:

$$y_{vest} = \begin{bmatrix} y_{scc} & y_{oto} & \beta_x & \beta_y & \beta_z & p_x & v_x & p_y & v_y & p_z & v_z & \dot{\beta}_x & \dot{\beta}_y & \dot{\beta}_z \end{bmatrix}^T \tag{3.26}$$

3.1.3 Matrici del sistema serie

Ricavati i modelli in spazio di stato relativi al sistema meccanico (3.6 e 3.7) ed al sistema vestibolare (3.24 e 3.25), discretizzato quest'ultimo si può procedere alla sintesi del sistema dato dalla serie dei due. Le matrici risultanti sono le seguenti (per semplicità non si distinguono nella notazione le matrici del sistema vestibolare continue da quelle discretizzate):

$$A_{serie} = \begin{bmatrix} A_m & 0 \\ B_{vest}\, C_m & A_{vest} \end{bmatrix} \qquad B_{serie} = \begin{bmatrix} B_m \\ B_{vest}\, D_m \end{bmatrix} \tag{3.27}$$

$$C_{serie} = \begin{bmatrix} D_{vest}\, C_m & C_{vest} \end{bmatrix} \qquad D_{serie} = \begin{bmatrix} D_{vest} & D_m \end{bmatrix} \tag{3.28}$$

Da cui si ricava il modello finale su cui eseguire l'MPC:

$$\Sigma_{serie} = (A_{serie}, B_{serie}, C_{serie}, D_{serie}) \tag{3.29}$$

3.2 Modello disaccoppiato

Il modello complessivo a sei gradi di libertà 3.29 ottenuto nel paragrafo precedente è di dimensione notevole e quindi può risultare problematico applicare l'MPC direttamente su di esso, data la mole dei tempi di calcolo richiesti dal controllo predittivo. Tuttavia osservando attentamente la struttura delle matrici relative al sistema meccanico ed al sistema vestibolare (3.6 e 3.7, 3.24 e 3.25) è immediato accorgersi che per quanto riguarda il sistema meccanico i sei gradi di libertà sono completamente disaccoppiati (la matrice A_m è diagonale) mentre per quanto concerne il sistema vestibolare è possibile suddividerlo in quattro sottosistemi l'uno indipendente dall'altro:

1. Σ_1, relativo a ψ;
2. Σ_2, relativo ad a_z;
3. Σ_3, relativo alla coppia a_x, θ;
4. Σ_4, relativo alla coppia a_y, ϕ.

in quanto la linearizzazione introdotta nella tilt coordination (3.17) ha ridotto il sistema complessivo ai due accoppiamenti indipendenti accelerazione longitudinale lungo x, posizione angolare di pitch (a_x, θ) e accelerazione longitudinale lungo y, posizione angolare di roll (a_y, ϕ), oltre ad aver separato la dinamica dell'accelerazione in z da quella delle rotazioni attorno agli altri due assi.
I quattro sottosistemi individuati possono essere messi ciascuno in serie al corrispettivo sistema meccanico producendo quattro differenti sistemi complessivi su cui eseguire l'MPC, disaccoppiati l'uno dall'altro, ciascuno dei quali può essere implementato su uno specifico calcolatore ottenendo così quattro calcolatori diversi che operano in parallelo efficientizzando lo svolgimento del problema e riducendo i tempo di calcolo.
L'analisi che segue si è quindi concentrata sui sistemi Σ_3, Σ_4 che rappresentano più degli altri due il fulcro dell'algoritmo di motion cueing; infatti il grado di libertà a_z (come facilmente intuibile) non è particolarmente sollecitato, mentre l'angolo di yaw ψ (legato quindi a ω_z) è molto importante per seguire visivamente l'andamento del veicolo nello schermo, ma non presenta particolari difficoltà nell'inseguire un riferimento in velocità che si mantenga entro i limiti massimi di mobilità della piattaforma. Nello specifico quindi si è proceduto realizzando il sistema Σ_3 (che d'ora in poi sarà indicato con la notazione Σ_{xy}) e le simulazioni e le verifiche sperimentali effettuate si sono concentrate su di esso. Infatti il comportamento relativo a Σ_4 sarà del tutto duale a meno di limiti e riferimenti diversi, con conseguente taratura.

3.2.1 Accoppiamento accelerazione x, angolo di pitch θ

Come per la sezione 3.1, per creare il sistema disaccoppiato si è proceduto innanzitutto realizzando il sistema meccanico che rappresenti il passo unitario di ritardo

di attuazione dei segnali di controllo nella piattaforma:

$$\Sigma_{m_{xy}} = (A_{m_{xy}}, B_{m_{xy}}, C_{m_{xy}}, D_{m_{xy}}) \tag{3.30}$$

con

$$A_{m_{xy}} = \begin{bmatrix} A_{a_x} & 0_{2\times2} \\ 0_{2\times2} & A_{\dot{\omega}_y} \end{bmatrix} \quad B_{m_{xy}} = \begin{bmatrix} B_{a_x} & 0_{2\times1} \\ 0_{2\times1} & B_{\dot{\omega}_y} \end{bmatrix} \quad C_{m_{xy}} = \begin{bmatrix} C_{a_x} & 0_{1\times2} \\ 0_{1\times2} & C_{\dot{\omega}_y} \end{bmatrix} \quad D_{a_{xy}} = 0_{2\times2}$$

Il modello del sistema vestibolare può essere ricavato in maniera altrettanto rapida, specialmente per i canali semicircolari, ma per non generare confusione con le notazioni e con le dimensioni delle matrici coinvolte, si preferisce illustrare nel dettaglio i singoli passaggi. Per i canali semicircolari, dal momento che ci si riduce ad un solo asse, si ha $y_{scc_y} = \hat{\omega}_y$, $u_{scc_y} = \omega_y = \dot{\theta}$ e dunque le matrici risultano:

$$A_{scc_y} = A_{s_y} \quad B_{scc_y} = B_{s_y} \quad C_{scc_y} = C_{s_y} \quad D_{scc_y} = D_{s_y} \tag{3.31}$$

Per le macule bisogna fare attenzione all'inserimento dell'azione di tilt coordination. La 3.18 per il solo asse x diventa infatti:

$$f_x = H \begin{bmatrix} \beta_y \\ a_x \end{bmatrix} = \begin{bmatrix} g & 1 \end{bmatrix} \begin{bmatrix} \theta \\ a_x \end{bmatrix} \tag{3.32}$$

Procedendo come nella sezione **??**, moltiplicando la matrice B_{oto_f} per la matrice H e partizionando la matrice risultante come indicato nella 3.19, procedendo poi all'inclusione della tilt coordination nel modello (3.1.2, 3.15), ridotto per un solo asse ad una forma analoga alla 3.31, si è finalmente in grado di costruire correttamente le matrici del sistema delle macule nel caso disaccoppiato, che forniscono $y_{oto_x} = \hat{f}_x$:

$$A_{oto_x} = \left[\begin{array}{c|c} A_{o_x} & B_2 \\ \hline 0_{1\times2} & 0 \end{array}\right] \quad B_{oto_x} = \begin{bmatrix} B_1 & 0_{2\times1} \\ 0 & 1 \end{bmatrix} \quad x_{oto_x} = \begin{bmatrix} x_x \\ \theta \end{bmatrix} \quad u_{oto_x} = \begin{bmatrix} a_x \\ \dot{\theta} \end{bmatrix} \tag{3.33}$$

e

$$C_{oto_x} = \begin{bmatrix} C_{o_x} & 0 \end{bmatrix} \quad D_{oto_x} = \begin{bmatrix} D_{o_x} & 0 \end{bmatrix} \tag{3.34}$$

Si può ora procedere alla costruzione del sistema vestibolare complessivo, mediante introduzione degli stati integrali la cui dinamica complessiva è stata descritta in 3.22 e che risulta di semplificazione immediata. Le matrici risultano:

$$\Sigma_{vest_{xy}} = (A_{vest_{xy}}, B_{vest_{xy}}, C_{vest_{xy}}, D_{vest_{xy}}) \tag{3.35}$$

con

$$A_{vest_{xy}} = \begin{bmatrix} A_{scc_y} & 0_{2\times3} & 0_{2\times2} \\ 0_{3\times3} & A_{oto_x} & 0_{3\times2} \\ 0_{2\times3} & 0_{2\times3} & A_{e_{xy}} \end{bmatrix} \quad B_{vest_{xy}} = \begin{bmatrix} 0_{2\times1} & B_{scc_y} \\ & B_{oto_x} \\ & B_{e_{xy}} \end{bmatrix} \tag{3.36}$$

e

$$C_{vest_{xy}} = \begin{bmatrix} C_{scc_y} & 0_{1\times3} & 0_{1\times2} \\ 0_{1\times2} & C_{oto_x} & 0_{1\times2} \\ 0_{3\times2} & 0_{3\times2} & I_{3\times3} \\ & 0_{1\times7} & \end{bmatrix} \quad D_{vest_{xy}} = \begin{bmatrix} 0 & D_{scc_y} \\ & D_{oto_x} \\ & 0_{3\times2} \\ 0 & 1 \end{bmatrix} \quad x_{vest_{xy}} = \begin{bmatrix} x_{scc_y} \\ x_{oto_x} \\ p_x \\ v_x \end{bmatrix} \tag{3.37}$$

avendo utilizzato le matrici

$$A_{e_{xy}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \quad B_{e_{xy}} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{3.38}$$

Le variabili d'ingresso sono $u = [a_x \ \dot{\theta}]^T$, mentre l'uscita è il vettore seguente, composto da sei elementi:

$$y_{vest_{xy}} = \begin{bmatrix} \hat{\omega}_y & \hat{f}_x & \theta & p_x & v_x & \dot{\theta} \end{bmatrix}^T \tag{3.39}$$

Il sistema serie complessivo relativo all'accoppiamento a_x, θ su cui si esegue l'MPC in questo lavoro di tesi sarà quindi dato da:

$$\Sigma_{serie_{xy}} = (A_{serie_{xy}}, B_{serie_{xy}}, C_{serie_{xy}}, D_{serie_{xy}}) \tag{3.40}$$

con:

$$A_{serie_{xy}} = \begin{bmatrix} A_{m_{xy}} & 0_4 \\ B_{vest_{xy}} C_{m_{xy}} & A_{vest_{xy}} \end{bmatrix} \qquad B_{serie_{xy}} = \begin{bmatrix} B_{m_{xy}} \\ B_{vest_{xy}} D_{m_{xy}} \end{bmatrix} \tag{3.41}$$

e

$$C_{serie_{xy}} = \begin{bmatrix} D_{vest_{xy}} C_{m_{xy}} & C_{vest_{xy}} \end{bmatrix} \qquad D_{serie_{xy}} = \begin{bmatrix} D_{vest_{xy}} & D_{m_{xy}} \end{bmatrix} \tag{3.42}$$

In realtà nel modello 3.40 non va inserito il modello vestibolare 3.35, 3.36, 3.37 ma la sua discretizzazione, dal momento che la serie va fatta tra modelli o entrambi discreti o entrambi continui e quello meccanico è discreto per costruzione. Le 3.41 e 3.42 presentano al loro interno la notazione continua del modello vestibolare unicamente per non appesantire la notazione introducendo ulteriori pedici, ma con queste righe si è sottolineato a dovere il fatto di dover procedere alla discretizzazione del sistema 3.35 prima di procedere alla costruzione della serie 3.40. Su quest'ultima, come detto, si va ad eseguire l'MPC; essa ha solo due gradi di libertà (cioè i due ingressi a_x e $\dot{\omega}_y$) e questo, unito alle minori dimensione del modello, riduce in modo sensibile i tempi di calcolo, specialmente in termini di ricerca dell'ingresso ottimo per il problema QP.

Capitolo 4

Analisi e Simulazioni effettuate

In questo capitolo si illustreranno le analisi effettuate sul controllo mediante MPC del sistema in esame. Si procederà per passi cercando di ricostruire il percorso che ci ha permesso di capire molte delle dinamiche e delle problematiche del sistema in esame, fornendo poi delle soluzioni ad alcuni di questi problemi con dei risultati significativi verificabili attraverso le numerose simulazione effettuate. A tale proposito, grazie alla collaborazione con VI-Grade, è stato possibile disporre di dati telemetrici completi relativi ad alcune sessioni di guida, da parte di un pilota dilettante, sui circuiti di Silverstone (GB, versione Grand Prix modificata nel 2010, figura 4.1) e Barcellona(E).

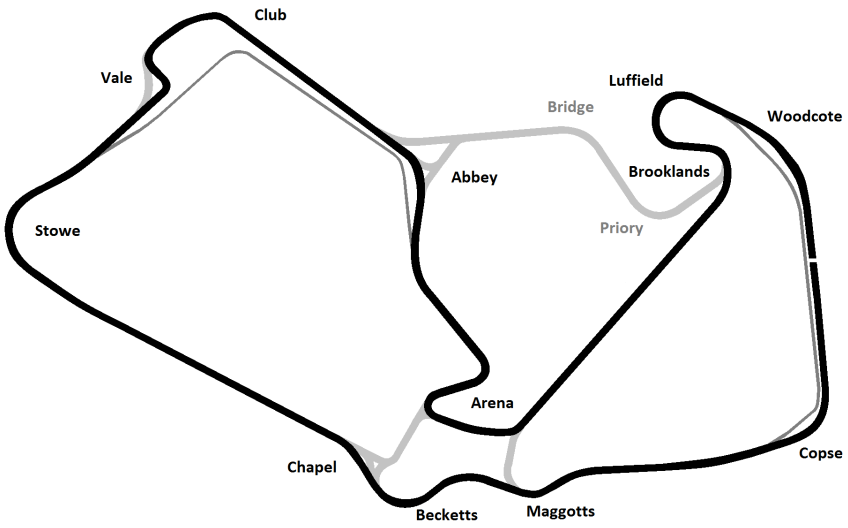


Figura 4.1: planimetria del circuito di Silverstone

Si è deciso di utilizzare i dati provenienti dal circuito inglese, per continuità con la tesi precedente di Daniele d'Ambrosio [1]. Data la mole di dati ed il tempo di campionamento abbastanza ridotto ($0.01s$) si è deciso di testare il controllo solamente

sul primo terzo del tracciato ($\cong 38s$), scelta che non è affatto restrittiva in quanto si riesce a testare il sistema in almeno due frenate significative della pista anche molto vicine tra loro. Inoltre confrontando le figure 4.1 e 4.2, si vede come l'andamento dell'accelerazione longitudinale, ovvero il riferimento principale del lavoro svolto, è abbastanza "regolare" nella'arco di un giro di pista: sono presenti le vistose frenate in corrispondenza delle curve, seguite da accelerazioni molto forti, disturbate nell'andamento dai cambi di marcia. Nel dettaglio osservando il riferimento in accelerazione longitudinale adoperato, figura 4.3, si vede come esso comprende il tratto fino all'uscita dalla curva Stowe. Esso dunque presenta la prima curva Copse, per la quale è richiesta una frenata non necessariamente impegnativa nonostante l'alta velocità di ingresso, il successivo allungo fino alle curve in rapida successione Maggotts, Becketts e Chapel, note a tutti gli appassionati di Formula 1 per la velocità con cui vanno percorse ed i repentini cambi di direzione che comportano, il rettilineo dell'Hangar Straight che porta alla suddetta curva Stowe, che nonostante sia una curva abbastanza di appoggio segue un lungo rettilineo e dunque è necessaria una decisa frenata prima di affrontarla. La porzione di riferimento si chiude con l'allungo che porta alla curva Vale.

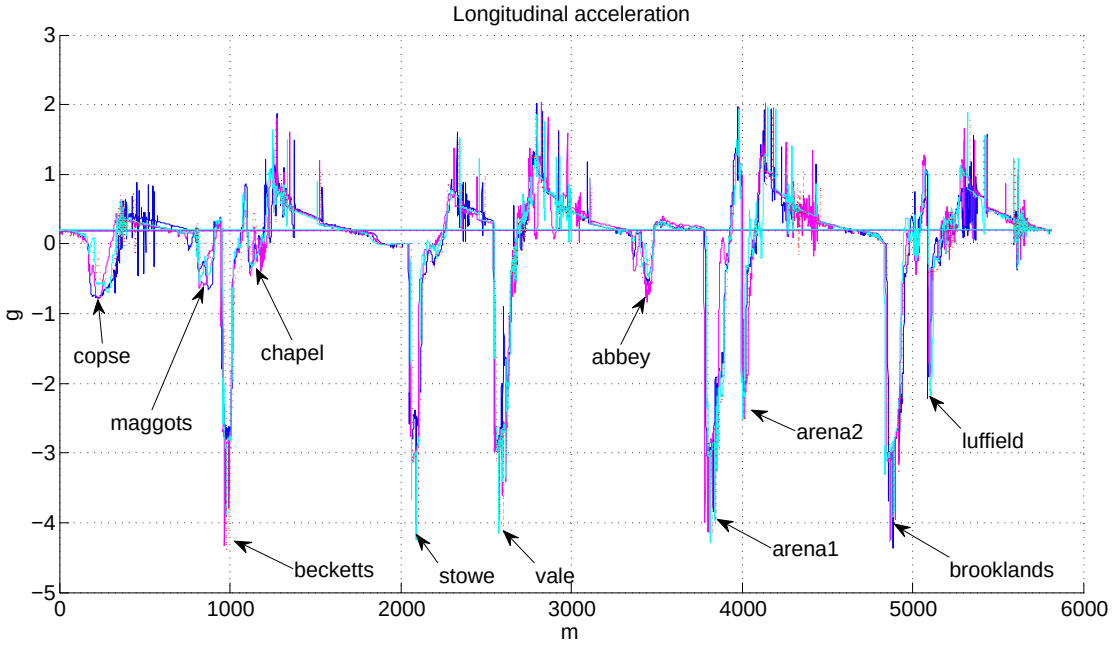


Figura 4.2: andamento delle accelerazioni per alcuni giri di pista a Silverstone, espressi in coordinate spazio.

In conclusione, il riferimento adottato presenta caratteristiche abbastanza varie (frenate impegnative e curve in appoggio, oltre a lunghi rettilinei) e ben si presta ad essere utilizzato come banco di prova per l'algoritmo di motion cueing sviluppato.

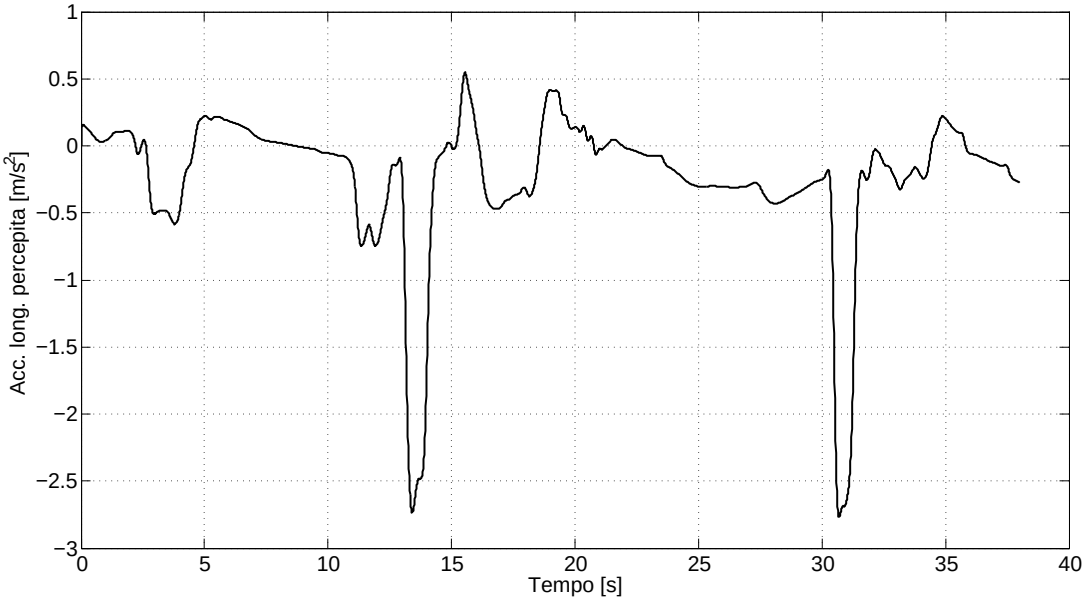


Figura 4.3: primi 38s del riferimento di acc. long. percepita

Tutte le simulazioni e le prestazioni in termini di tempo di calcolo, sono state ottenute sul mio notebook che ha processore Intel Core 2 da 2.10GHz con 4GB di Ram e sistema operativo a 64bit.

4.1 Analisi dell'importanza della predizione

In questa prima sezione si analizzerà nel dettaglio il ruolo dell'orizzonte di predizione nelle prestazioni del controllo proposto. Come anticipato nella sezione questo parametro, intuitivamente già di per se molto importante per un sistema di controllo basato su MPC, risulta determinante sia in termini di qualità di inseguimento del riferimento sia in termini di tempo di calcolo, tuttavia assume un significato molto diverso a seconda che si faccia predizione in modo intelligente (con *look ahead* o la strategia di macthing proposta) oppure con strategie più elementari (come *senza look ahead*).
Il set di partenza dei parametri utilizzato per l'MPC è quello riportato nelle tabelle seguenti. I limiti del sistema sono quelli nominali mentre la configurazione dei pesi utilizzata è il risultato di una taratura robusta (e quindi anche abbastanza conservativa soprattutto in pitch) a cui si è giunti dopo numerosi test. Si noti come però, in termini di velocità di pitch applicata in ingresso (ω_y) e velocità di pitch percepita ($\hat{\omega}_y$), si sono ulteriormente limitati i valori nominali per limitare l'effetto della tilt coordination in termini di percezione delle rotazioni. Per quanto concerne i pesi

Uscita	Peso	Limite inf.	Limite sup.
$\hat{w}_y$	0	-1.06/6	1.06/6
$\hat{f}_x$	0.1	-inf	+inf
β_y	1	-0.26	0.26
p_x	0.5	-0.5	0.5
v_x	0.3	-1.3	1.3
$\dot{\beta}_y$	0	-1.06	1.06

Tabella 4.1: pesi e limiti sulle sei uscite del MPC

Ingresso	Peso	Limite inf.	Limite sup.
Δf_x	10^0	-3.3/10	3.3/10
Δw_y	10^0	-1.06/10	1.06/10
f_x	10^{-5}	-3.3	3.3
w_y	10^{-2}	-1.06/2	1.06/2

Tabella 4.2: pesi e limiti sugli ingressi del MPC

ovviamente nelle simulazioni che seguiranno, in funzione della lunghezza dell'intervallo di predizione e del tipo di generazione di riferimento considerata, sono stati leggermente modificati per adattarli al contesto di utilizzo, tuttavia la logica della taratura non cambia in modo significativo. Si noti come i pesi sugli ingressi assoluti siano dell'ordine di 10^{-5} per l'accelerazione longitudinale applicata e di 10^{-2} per la velocità di pitch (per favorire l'inseguimento del riferimento utilizzando nello spazio la piattaforma) mentre quelli relativi alla variazioni degli stessi sono dell'ordine di 10^0 per evitare brusche variazioni del segnale di ingresso da un istante all'altro (con corrispettive oscillazioni indesiderate nel jerk percepito, parametro sulla cui importanza ha ampiamente indagato il collega Mauro Pozzi nel suo lavoro di tesi [2]). Particolarmente significativo anche il peso dato al riferimento in accelerazione longitudinale percepita il cui inseguimento, come rimarcato più volte, è il principale obiettivo del motion cueing; esso è solo dell'ordine di 10^{-1} ed è addirittura inferiore ai pesi attribuiti alle uscite di posizione e velocità longitudinale e angolo di pitch. Questo settaggio, in parte controintuitivo, è indice del fatto che la taratura di un sistema di controllo bastato su MPC non è sempre così facile.

4.1.1 Grafici delle simulazioni

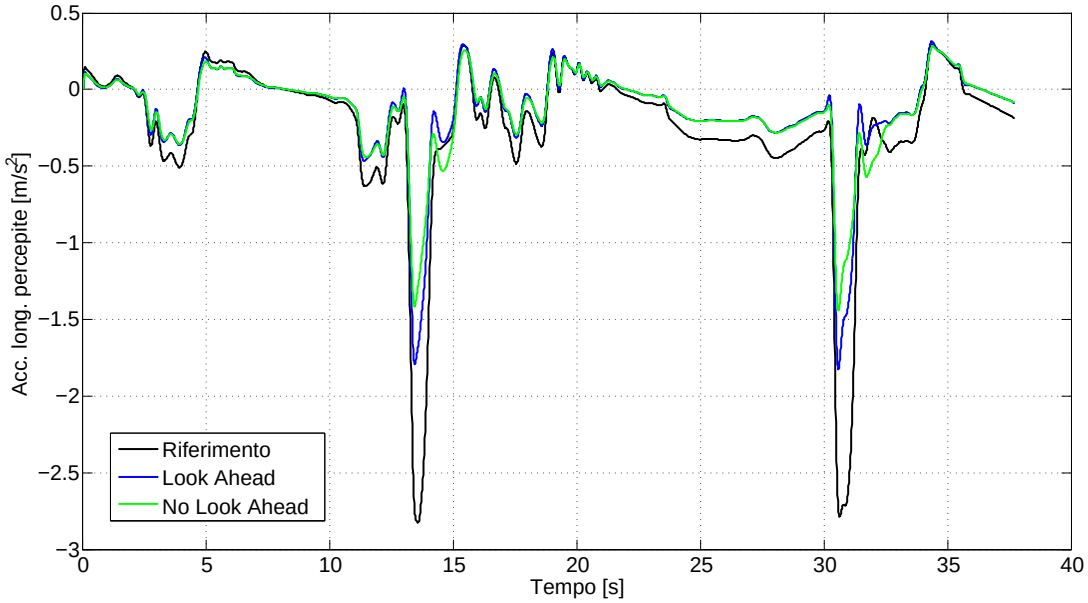


Figura 4.4: confronto tra le acc.long. percepite con o senza Look Ahead, $T_{pred} = 0.3s$

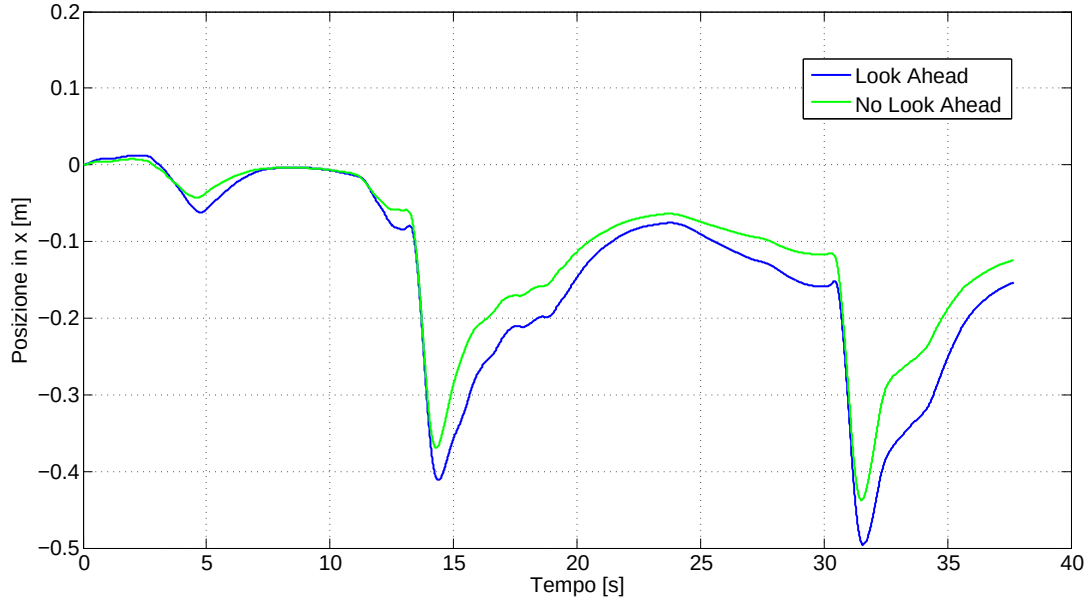


Figura 4.5: confronto della pos. long. con o senza Look Ahead, $T_{pred} = 0.3s$

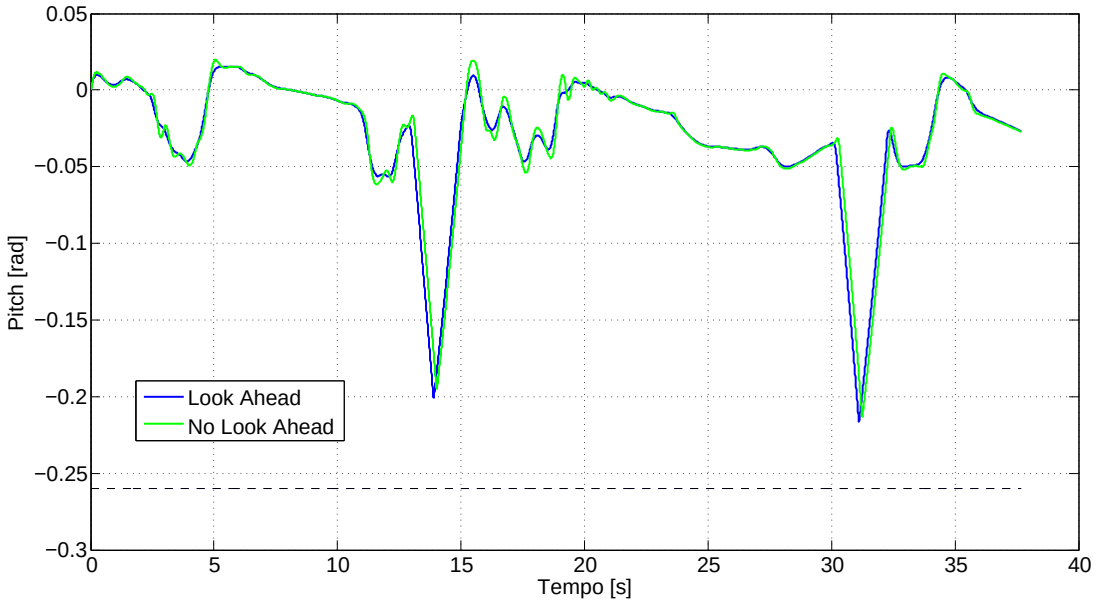


Figura 4.6: confronto del pitch con o senza Look Ahead, $T_{pred} = 0.3s$

Come è possibile osservare da questi primi grafici le prestazioni, in termini di inseguimento del riferimento in accelerazione longitudinale percepita, con una predizione di soli 3 decimi sono molto simili con o senza look ahead. L'inseguimento con look ahead è più efficace essenzialmente nelle due frenate e questo si giustifica con un più ampio sfruttamento della piattaforma in termini di spostamenti longitudinali. Si noti invece come lo sfruttamento del pitch nei due casi sia praticamente lo stesso, con un leggero ritardo da parte di quello senza look ahead come era prevedibile aspettarsi.
Anche con $T_{pred} = 0.5s$ (figura 4.7) le prestazioni sono abbastanza simili e, come detto in precedenza, questo valore rappresenta la soglia oltre il quale il controllo senza look ahead non è più efficace, mentre l'importanza dell'avere una buona predizione inizia a manifestarsi. Nelle figure 4.8, 4.9, 4.10 ottenute con $T_{pred} = 0.8s$ è proprio possibile osservare quanto detto; non solo in termini di inseguimento del riferimento ma anche di sfruttamento della piattaforma (soprattutto longitudinalmente). Infine gli ultimi grafici sono relativo ad un tempo di predizione pari a $1.5s$ utilizzando la strategia con look ahead: risulta evidente come in questo caso lo sfruttamento della piattaforma sia ottimale e senza un utilizzo eccessivo del pitch; il riferimento è inseguito perfettamente il che suggerisce come per il problema in questione, un'orizzonte di predizione che possa considerarsi ampiamente adeguato sia dell'ordine di $1.5s - 2s$.

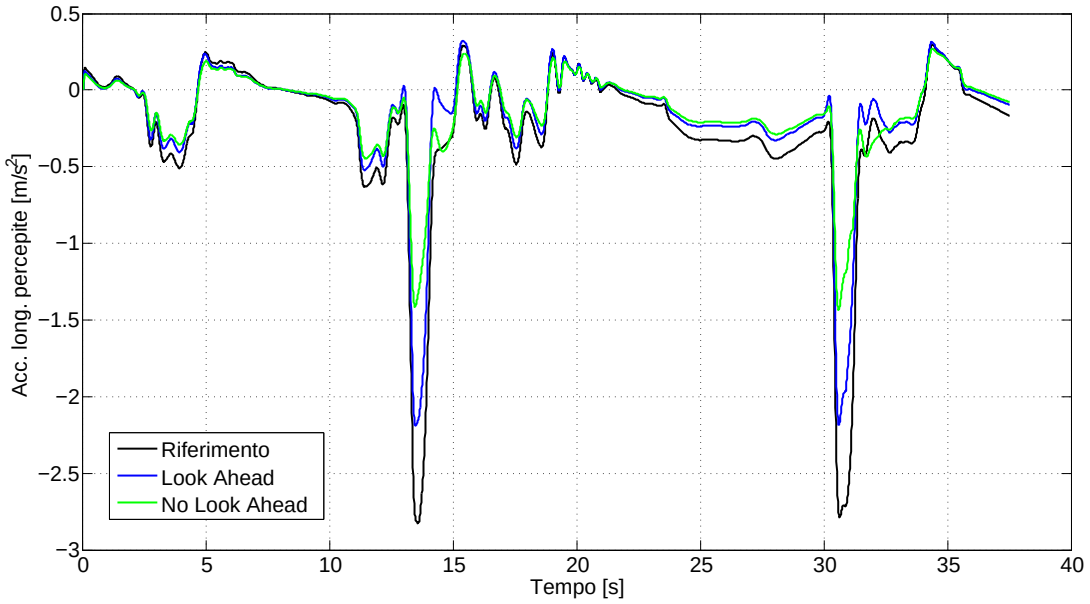


Figura 4.7: confronto tra le acc.long. percepite con o senza Look Ahead, $T_{pred} = 0.5s$

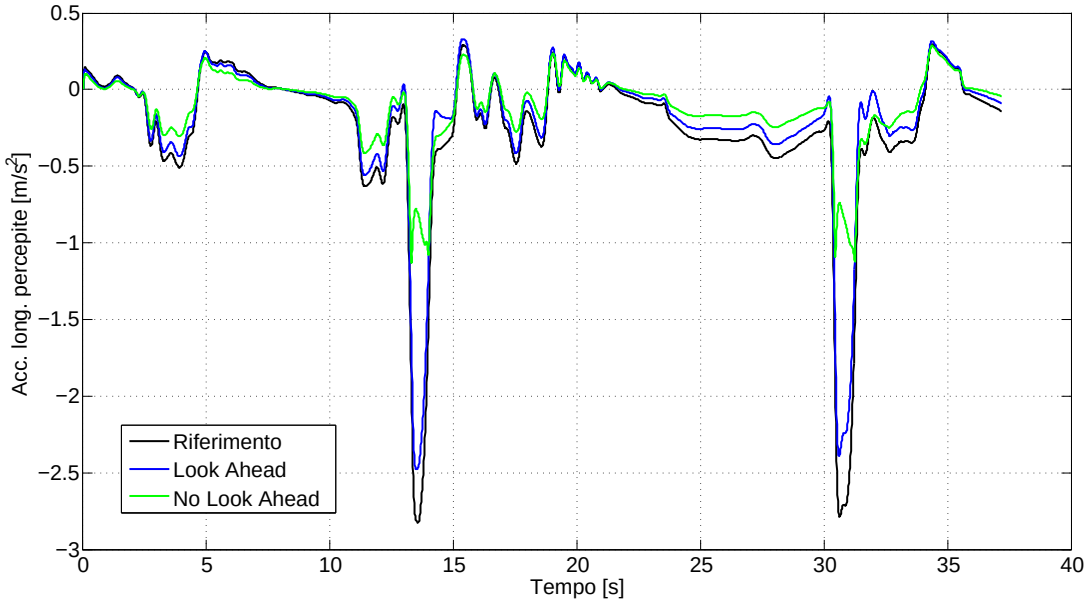


Figura 4.8: confronto tra le acc.long. percepite con o senza Look Ahead, $T_{pred} = 0.8s$

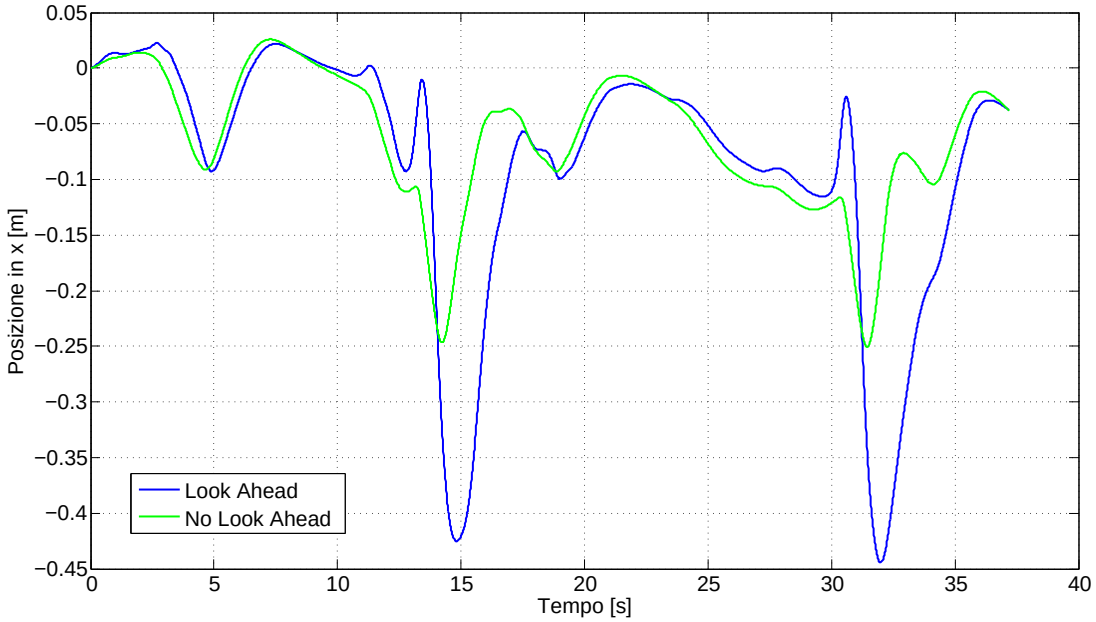


Figura 4.9: confronto della pos. long. con o senza Look Ahead, $T_{pred} = 0.8s$

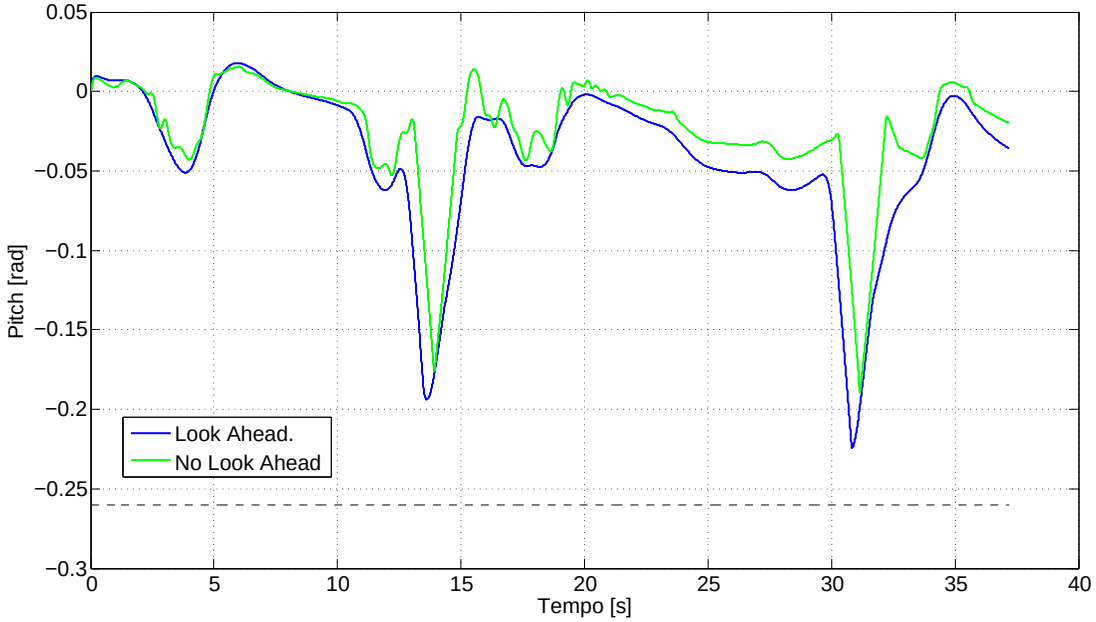


Figura 4.10: confronto del pitch con o senza Look Ahead, $T_{pred} = 0.8s$

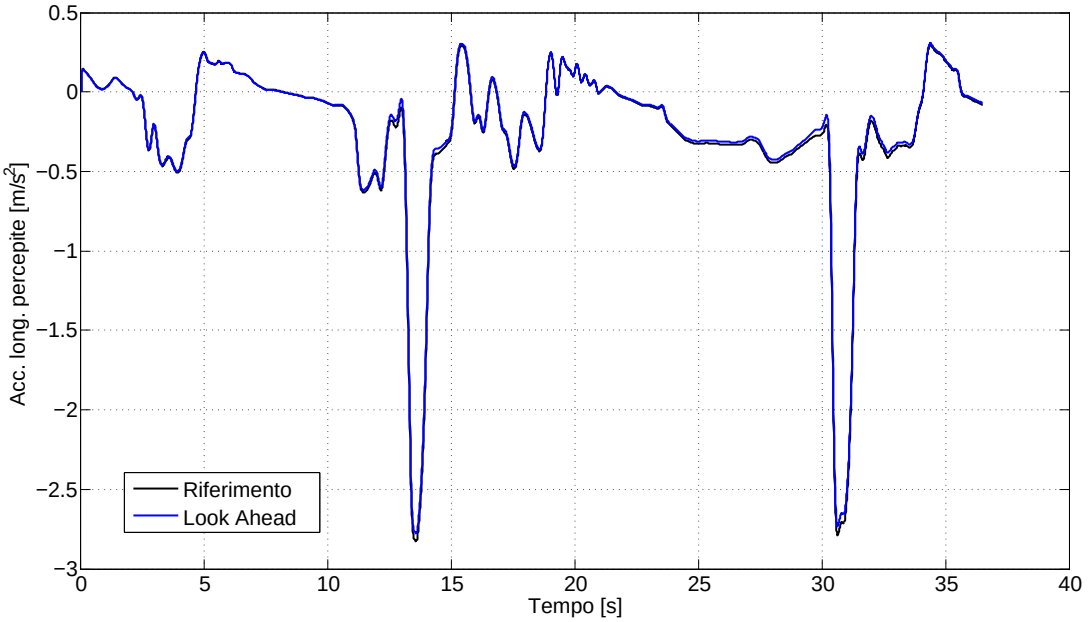


Figura 4.11: accelerazione longitudinale percepita con Look Ahead, $T_{pred} = 1.5s$

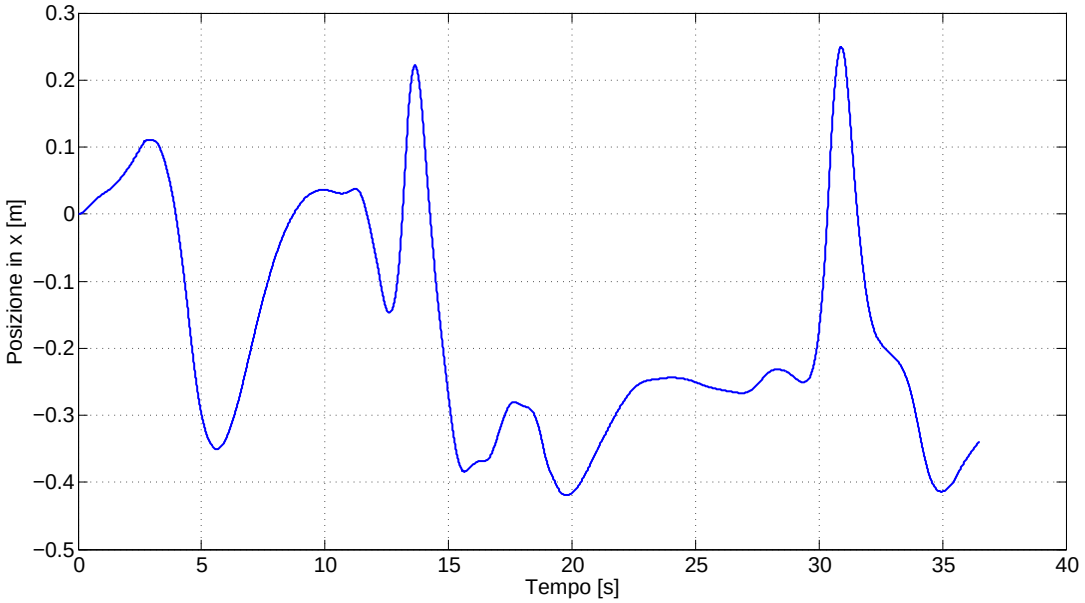


Figura 4.12: posizione longitudinale con Look Ahead, $T_{pred} = 1.5s$

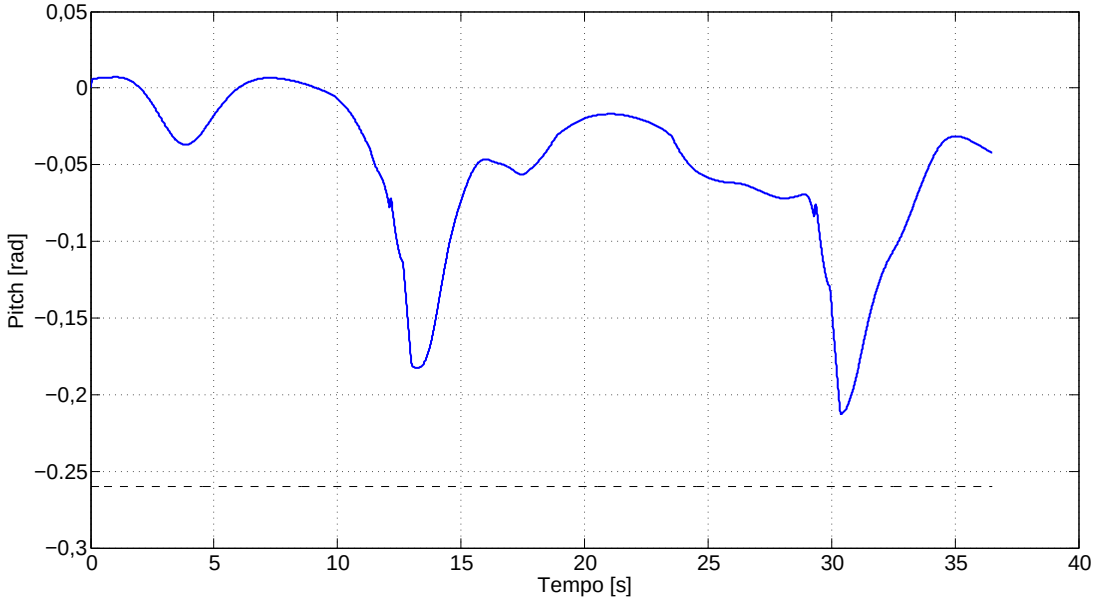


Figura 4.13: pitch con Look Ahead, $T_{pred} = 1.5s$

	Tpred	**Tempo di simulazione**	**Norma errore**
Look Ahead	0.3s	25s	14.68
No Look Ahead	0.3	30s	18.39
Look Ahead	0.5s	41s	10.37
No Look Ahead	0.5	46s	17.82
Look Ahead	0.8s	132s	6.89
No Look Ahead	0.8	137s	21.22
Look Ahead	1.5s	476s	1.22

Tabella 4.3: prestazioni con o senza Look Ahead, T_{pred} variabile

Analizzando i grafici e le prestazioni riportate nella tabella risulta quindi evidente che i vantaggi dell'avere un orizzonte di predizione sufficientemente ampio, con una predizione adeguata, sono significativi in quanto lo sfruttamento della piattaforma (e quindi l'inseguimento del riferimento) è ottimale, però si paga in termini di tempo di calcolo in modo considerevole. Riuscire a mantenere elevato l'orizzonte di predizione, diminuendo l'onere computazionale, sarà quindi l'obiettivo della decimazione della predizione, argomento trattato nella prossima sezione.

4.2 Decimazione della predizione

Il principale limite all'utilizzo online del MPC per il controllo del sistema in esame, è il tempo di calcolo richiesto. Infatti, nonostante l'utilizzo di un ottimizzatore adeguato ed una implementazione efficiente, l'onere computazione aumenta notevolmente al crescere dell'orizzonte di predizione. Tuttavia l'efficacia del MPC è strettamente legata alla possibilità di disporre di un riferimento adeguato e di un orizzonte di predizione che sia appropriato per le dinamiche del sistema in questione; nello specifico, come è emerso nella sezione precedente, si è individuato in un $1s$, $1.5s$ una lunghezza di tale parametro adeguata per sfruttare al meglio le potenzialità della piattaforma nel rendere il motion cueing. Pertanto la necessità di disporre di un orizzonte di predizione di tale lunghezza mantenendo al contempo i tempi di calcolo dell'ordine dei secondi di giro simulati, ci ha spinti a modificare l'MPC classico adottando una decimazione della predizione.

4.2.1 Estensione del MPC

La frequenza base a cui opera il sistema in esame è di $100Hz$; ciò significa che ogni $0.01s$ il sistema invia i segnali di controllo per la piattaforma. L'algoritmo classico di MPC implementato, considerando quindi un periodo T_s per la discretizzazione del modello pari a $0.01s$, genera sulla predizione un numero di campioni N_p pari a:

$$N_p = \frac{T_{pred}}{T_s}$$

e sulla base di questo numero si creano le matrici F, Φ della predizione e quelle dei vincoli A, b, rispetto alle quali poi opera l'algoritmo di ottimizzazione (come spiegato in 2). Il tempo di calcolo cresce notevolmente e in modo non lineare in funzione di N_p (supponendo $N_c = N_p$) e per mantenerlo dell'ordine della durata di simulazione si è constatato a livello simulativo che tale parametro deve essere dell'ordine di 40, 50 (coerente quindi con quanto esposto nel capitolo precedente in cui più volte si è sottolineato l'utilizzo di $T_{pred} \cong 0.5s$ come orizzonte di predizione significativo). Da queste analisi nasce l'idea di decimare l'orizzonte di predizione mantenendo comunque di $100Hz$ la frequenza con cui opera il sistema di controllo; l'idea di base non è solo quella di abbattere notevolmente i tempi di calcolo, ma è anche quella di verificare che per controllare in modo efficiente il sistema in esame è possibile ridurre a $50Hz$, $20Hz$ almeno la dinamica di predizione considerando tali frequenze sufficientemente elevate per non creare aliasing nell'interpretare il comportamento futuro del sistema. A tale proposito le simulazione che seguiranno da un lato mostreranno il vantaggio di adoperare un orizzonte di predizione maggiore (con decimazione) a parità di onere computazionale (e quindi di N_p) rispetto al caso base senza decimazione; in secondo luogo si mostrerà come la decimazione permetta di ridurre notevolmente i tempi di calcolo senza compromettere l'inseguimento del riferimento (a parità quindi di T_{pred}).

L'estensione dell'algoritmo classico di MPC proposta è del tutto generica e consiste nel dividere l'orizzonte di predizione in N intervalli, scegliendo per ognuno il passo di decimazione appropriato:

$$T = \begin{bmatrix} T_1 & T_2 & \dots & T_N \end{bmatrix}$$

con $T_N = T_{pred}$.
Rispettivamente si definisce K il vettore dei corrispettivi passi di decimazione:

$$K = \begin{bmatrix} K_1 & K_2 & \dots & K_N \end{bmatrix}$$

Con questa notazione si intende che ad esempio scegliendo dei vettori T, K come mostrato:

$$T = \begin{bmatrix} 0.2 & 0.6 & 1 \end{bmatrix}$$
$$K = \begin{bmatrix} 1 & 2 & 4 \end{bmatrix}$$

si riesce ad avere un orizzonte di predizione $T_{pred} = 1s$ con un $N_p \neq 100$, bensì modificato (detto d'ora in avanti $N_{p,eq}$ per distinguerlo dal caso base) e calcolabile come segue:

$$N_{p,eq} = \frac{T_1}{K_1} + \frac{T_2 - T_1}{K_2} + \dots + \frac{T_{pred} - T_N}{K_N} = N_1 + \dots + N_N = 40$$

come se si utilizzasse la procedura classica di MPC con $T_{pred} = 0.4s$.

Di seguito sono riportate le matrici F, Φ della predizione modificata solo (nel caso $N = 3$ per semplicità espositiva) supponendo di considerare Δu costante tra un passo di decimazione e il successivo. Capita la logica della struttura delle matrici, l'estensione al caso generale implementato è immediata.

$$F = \begin{bmatrix} CA^{k_1} \\ \vdots \\ CA^{N_1 k_1} \\ CA^{N_1 k_1 + k_2} \\ CA^{N_1 k_1 + 2k_2} \\ \vdots \\ CA^{N_1 k_1 + N_2 k_2} \\ CA^{N_1 k_1 + N_2 k_2 + k_3} \\ \vdots \\ CA^{N_1 k_1 + N_2 k_2 + N_3 k_3} \end{bmatrix} \tag{4.1}$$

$$
\left[\begin{array}{c|ccc|ccc}
CB & 0 & \dots & 0 & 0 & \dots & 0 \\
CAB & CB & \dots & 0 & 0 & \dots & 0 \\
\dots & & & \dots & & & \\
CA^{N_1-1}B & CA^{N_1-2}B & \dots & CB & 0 & & 0 \\
\hline
CA^{N_1-1+k_2}B & CA^{N_1-2+k_2}B & \dots & CA^{k_2}B & C(A^{k_2-1}+\dots+A+I)B & & 0 \\
CA^{N_1-1+2k_2}B & CA^{N_1-2+2k_2}B & \dots & CA^{2k_2}B & CA^{k_2}(A^{k_2-1}+\dots+A+I)B & & C(A^{k_2-1}+\dots+A+I)B \\
\dots & & & \dots & \dots & & \\
CA^{N_1+N_2k_2-1}B & CA^{N_1+N_2k_2-2}B & \dots & CA^{N_2k_2}B & CA^{(N_2-1)k_2}(A^{k_2-1}+\dots+A+I)B & & CA^{(N_2-2)k_2}(A^{k_2-1}+\dots+A+I)B \\
\hline
CA^{N_1+N_2k_2-1+k_3}B & CA^{N_1+N_2k_2-2+k_3}B & \dots & CA^{N_2k_2+k_3}B & CA^{(N_2-1)k_2+k_3}(A^{k_2-1}+\dots+A+I)B & & CA^{(N_2-2)k_2+k_3}(A^{k_2-1}+\dots+A+I)B \\
\dots & & & \dots & \dots & & \\
CA^{N_1+N_2k_2+N_3k_3-1}B & CA^{N_1+N_2k_2+N_3k_3-2}B & \dots & CA^{N_2k_2+N_3k_3}B & CA^{(N_2-1)k_2+N_3k_3}(A^{k_2-1}+\dots+A+I)B & & CA^{(N_2-2)k_2+N_3k_3}(A^{k_2-1}+\dots+A+I)B
\end{array}\right]
$$

$$
\left[\begin{array}{ccc|ccc}
\dots & 0 & \dots & 0 & \dots & 0 \\
\dots & 0 & \dots & 0 & \dots & 0 \\
 & \dots & & \dots & & \dots \\
\dots & 0 & & 0 & & 0 \\
\hline
\dots & 0 & & 0 & \dots & 0 \\
\dots & 0 & & 0 & \dots & 0 \\
 & \dots & & \dots & & \dots \\
\dots & C(A^{k_2-1}+\dots+A+I)B & & 0 & \dots & 0 \\
\hline
\dots & CA^{k_3}(A^{k_2-1}+\dots+A+I)B & & C(A^{k_3-1}+\dots+A+I)B & \dots & 0 \\
 & \dots & & & & \dots \\
\dots & CA^{N_3k_3}(A^{k_2-1}+\dots+A+I)B & & CA^{(N_3-1)k_3}(A^{k_3-1}+\dots+A+I)B & \dots & C(A^{k_3-1}+\dots+A+I)B
\end{array}\right]
= \Phi \qquad (4.2)
$$

Necessitano una modifica anche le matrici dei vincoli A, b. Infatti (considerando per semplicità il caso con $k_1 = 1$, $k_2 = 2$, $k_3 = 3$, $N_1 = 2$, $N_2 = 2$ e $N_3 = 1$) la nuova struttura dell'ingresso è:

$$\begin{aligned}
u(k) &= \Delta u(k) + u(-1) \\
u(k+1) &= \Delta u(k) + \Delta u(k+1) + u(-1) \\
\Big(u(k+2) &= \Delta u(k) + \Delta u(k+1) + \Delta u(k+2) + u(-1)\Big) \\
u(k+3) &= \Delta u(k) + \Delta u(k+1) + \Delta u(k+2) + \Delta u(k+3) + u(-1) \\
&= \Delta u(k) + \Delta u(k+1) + 2\Delta u(k+3) + u(-1) \\
\Big(u(k+4) &= \Delta u(k) + \Delta u(k+1) + 2\Delta u(k+3) + \Delta u(k+4) + u(-1)\Big) \\
u(k+5) &= \Delta u(k) + \Delta u(k+1) + 2\Delta u(k+3) + \Delta u(k+4) + \Delta u(k+5) + u(-1) \\
&= \Delta u(k) + \Delta u(k+1) + 2\Delta u(k+3) + 2\Delta u(k+5) + u(-1) \\
&\vdots \\
u(k+8) &= \Delta u(k) + \Delta u(k+1) + 2\Delta u(k+3) + 2\Delta u(k+5) + 3\Delta u(k+8) + u(-1)
\end{aligned} \tag{4.3}$$

Da cui risulta evidente che non cambia la struttura dei vincoli in Δu (così come una volta modificato in modo opportuno le matrici F, Φ non cambiano i vincoli sulle uscite) bensì quella nell'ingresso u assoluto:

$$\begin{bmatrix} u(k) \\ u(k+1) \\ u(k+2) \\ u(k+3) \\ u(k+4) \end{bmatrix} = \begin{bmatrix} I & 0 & 0 & 0 & 0 \\ I & I & 0 & 0 & 0 \\ I & I & 2I & 0 & 0 \\ I & I & 2I & 2I & 0 \\ I & I & 2I & 2I & 3I \end{bmatrix} \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \Delta u(k+2) \\ \Delta u(k+3) \\ \Delta u(k+4) \end{bmatrix} + \begin{bmatrix} u(k-1) \\ u(k-1) \\ u(k-1) \\ u(k-1) \\ u(k-1) \end{bmatrix} \tag{4.4}$$

Anche in questo caso la generalizzazione dell'espressione appena scritta è immediata.

4.2.2 Simulazioni con decimazione

Come espresso precedentemente, in questa sezione si metteranno in luce i vantaggi dell'applicazione del MPC con l'estensione proposta al sistema in esame: si farà vedere come, a parità di onere computazionale e di tempo di simulazione quindi, l'aver decimato la predizione, permettendo quindi al controllo un T_{pred} maggiore, permetta di ottenere delle prestazioni migliori in termini di inseguimento del riferimento. Infine si evidenzierà come con la procedura proposta sia possibile adoperare T_{pred} dell'ordine di $1s$, $1.5s$ come desiderato, rendendo accettabili i tempi calcolo.

A parità di onere computazionale

Con questa espressione si intende che si confrontano l'MPC classico con la variante proposta nel caso in cui si prendano lo stesso numero di punti lungo l'orizzonte di

predizione, il che significa:

$$N_{p,eq} = N_p$$

coerentemente con le definizione date dei due parametri.

1° Esempio. In questo primo esempio si sono considerati i seguenti parametri:

$$T_{pred,noDec} = 0.3s \quad \Rightarrow \quad N_p = 30$$

per quanto concerne l'MPC classico senza decimazione.

$$T = \begin{bmatrix} 0.1 & 0.5 \end{bmatrix} \qquad K = \begin{bmatrix} 1 & 2 \end{bmatrix}$$

per quanto concerne l'algortimo con decimazione, il che comporta un

$$T_{Pred,Dec} = 0.5s$$

con ovviamente $N_{p,eq} = N_p$.

2° Esempio. Nel secondo esempio invece si sono considerati i seguenti parametri:

$$T_{pred,noDec} = 0.6s \quad \Rightarrow \quad N_p = 60$$

per quanto concerne l'MPC classico senza decimazione.

$$T = \begin{bmatrix} 0.2 & 0.6 & 1.4 \end{bmatrix} \qquad K = \begin{bmatrix} 1 & 2 & 4 \end{bmatrix}$$

per quanto concerne l'algortimo con decimazione, il che comporta un

$$T_{Pred,Dec} = 1.4s$$

con ovviamente $N_{p,eq} = N_p$.

Come sarà visibili dai grafici che seguono le prestazioni in termini di inseguimento del riferimento (4.14, 4.15), con o senza decimazione, sono simili ma leggermente a favore del caso con decimazione il che prova come, a parità di onere computazionale, avere un orizzonte di predizione maggiore favorisca il sistema di controllo nello sfruttamento della piattaforma. In termini di tempo di calcolo ovviamente i due sono essenzialmente equivalenti, come era giusto aspettarsi.

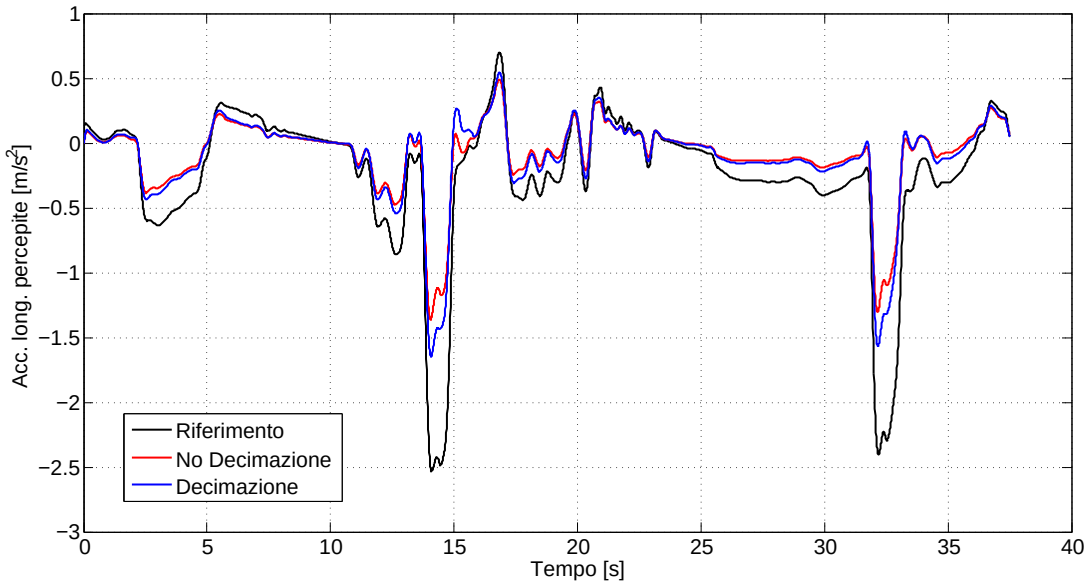


Figura 4.14: confronto delle acc. percepite con o senza decimazione, 1° Es.

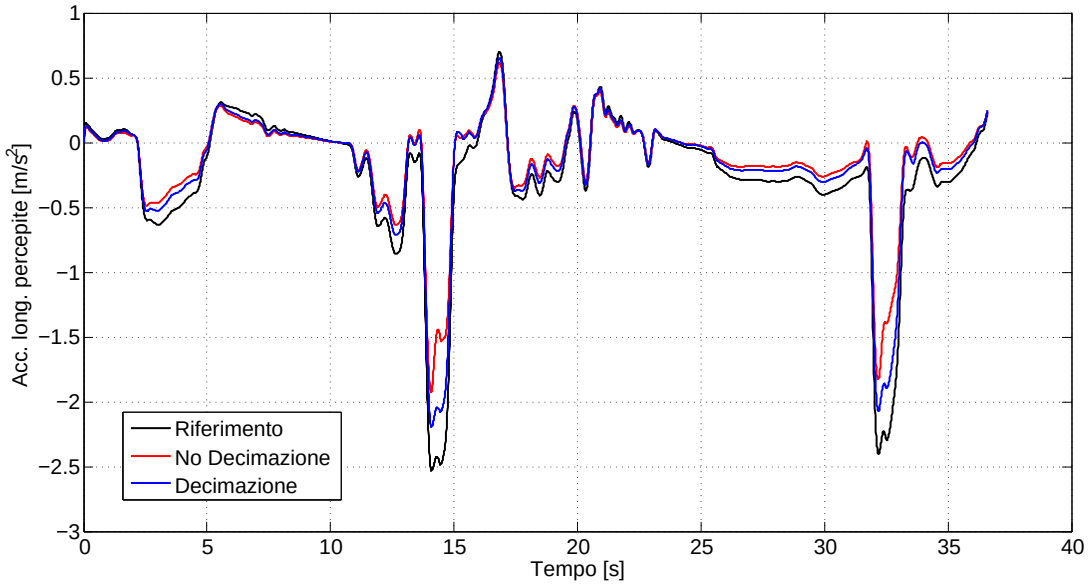


Figura 4.15: confronto delle acc. percepite con o senza decimazione, 2° Es.

	Tempo di simulazione	**Norma errore**
1°Es. No Decimazione	18s	19.06
1°Es. Decimazione	18s	16.22
2°Es. No Decimazione	58s	12.90
2°Es. Decimazion	56s	7.34

Tabella 4.4: prestazione esempi 1,2 a parità di onere computazionale

A parità di orizzonte di predizione

Questo tipo di prove hanno l'obiettivo di verificare che, decimando in modo appropriato l'orizzonte di predizione, si possono rendere i tempi di calcolo accettabili mantenendo quasi inalterata la qualità dell'inseguimento del riferimento; questo è essenzialmente il vantaggio fondamentale che offre l'estensione del MPC proposta. Anche in questo caso si riportano due esempi significativi, con relativi grafici e tabella delle prestazioni.

1° Esempio. Nel primo esempio si è utilizzato un $T_{pred} = 0.8s$ comune ad entrambi e si è utilizzato il seguente passo di decimazione:

$$T = \begin{bmatrix} 0.1 & 0.5 & 0.8 \end{bmatrix} \qquad K = \begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

il che comporta un $N_{p,eq} = 40$.

2° Esempio. Per il secondo esempio invece si è utilizzato un $T_{pred} = 1.5s$ comune ad entrambi e si è utilizzato il seguente passo di decimazione:

$$T = \begin{bmatrix} 0.4 & 1 & 1.5 \end{bmatrix} \qquad K = \begin{bmatrix} 2 & 3 & 5 \end{bmatrix}$$

il che comporta un $N_{p,eq} = 50$.

Come è possibile constatare osservando i grafici riportati e la tabella relativa alle prestazioni, il peggiormente in termini di inseguimento del riferimento, utilizzando la decimazione, è trascurabile mentre il guadagno in termini di tempi di calcolo è stato estremamente rilevante tale da produrre tempi di calcolo inferiori ai secondi di giri presi in esame con un orizzonte di predizione elevato ($T_{pred} = 0.8s, 1.5s$). Si noti infatti come il valore di $N_{p,eq}$ (il parametro a cui è effettivamente legato l'onere computazionale del MPC) sia 40 per il primo esempio e 50 nel secondo, dato coerente con quanto espresso più volte precedentemente circa il fatto che senza decimazione un $T_{pred} = 0.5s$ (quindi $N_p = 50$) era un orizzonte di predizione adeguato in termini di tempi di calcolo. Le prestazioni, in termini di inseguimento del riferimento, sono migliori nel set up del secondo esempio, ulteriore conferma del fatto che $1.5s$ sia un buon orizzonte di predizione per controllare efficacemente

il sistema in esame. Un'ultima considerazione va fatta in relazione al grafico 4.18; tale grafico è relativo allo sfruttamento longitudinale della piattaforma nel secondo esempio. L'utilizzo di un passo di decimazione più significativo ha comportato una certa differenza negli spostamenti della piattaforma tra i due casi (tale differenza non emerge in modo significativo nel primo esempio o nel pitch del secondo) tuttavia il comportamento in termini di inseguimento del riferimento è molto buono per entrambi. Una tale differenza nella scelta del controllo del sistema è ragionevole ed è dovuta al differente spazio di ottimizzazione in cui si opera con o senza decimazione che appunto nel secondo esempio è abbastanza diverso nei due casi.

	Tempo di simulazione	**Norma errore**
1°Es. No Decimazione	113s	10.11
1°Es. Decimazione	28s	11.47
2°Es. No Decimazione	442s	3.57
2°Es. Decimazione	37s	4.47

Tabella 4.5: prestazione esempi 1,2 a parità di T_{pred}

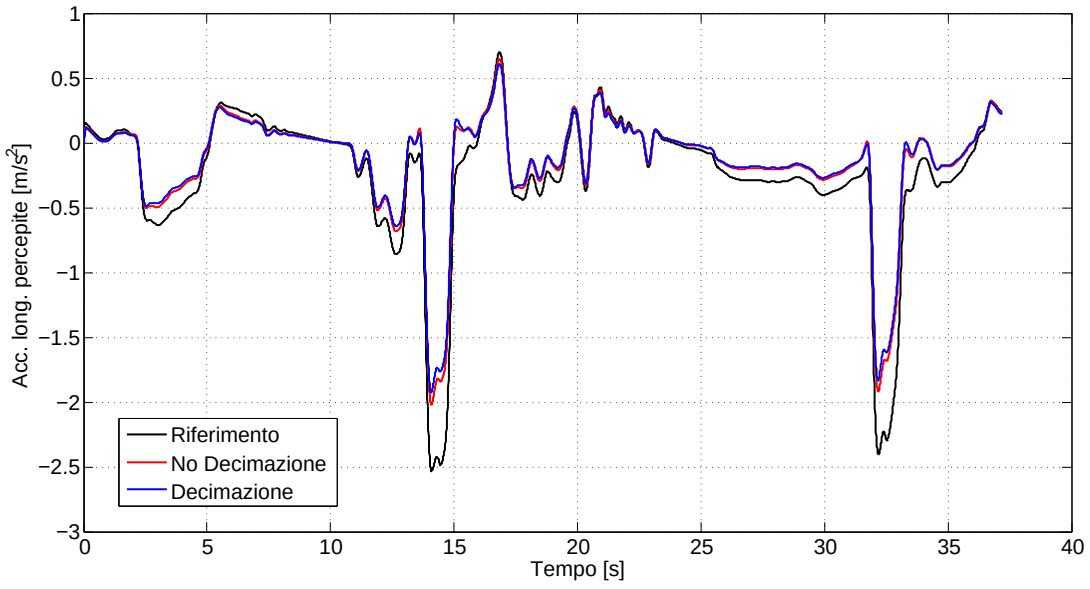


Figura 4.16: confronto delle acc. percepite con o senza decimazione, 1° Es.

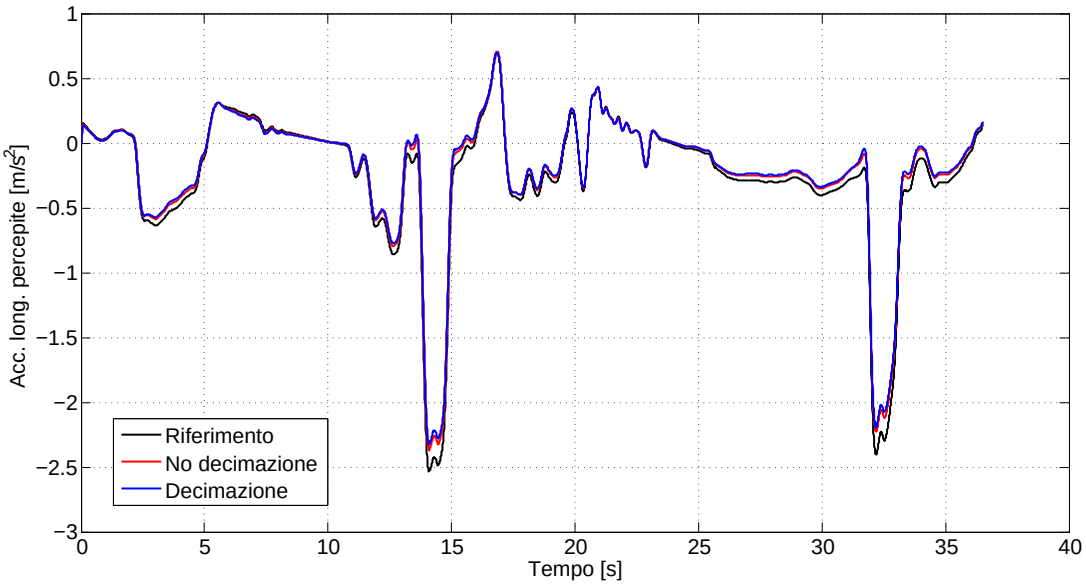


Figura 4.17: confronto delle acc. percepite con o senza decimazione, 2° Es.

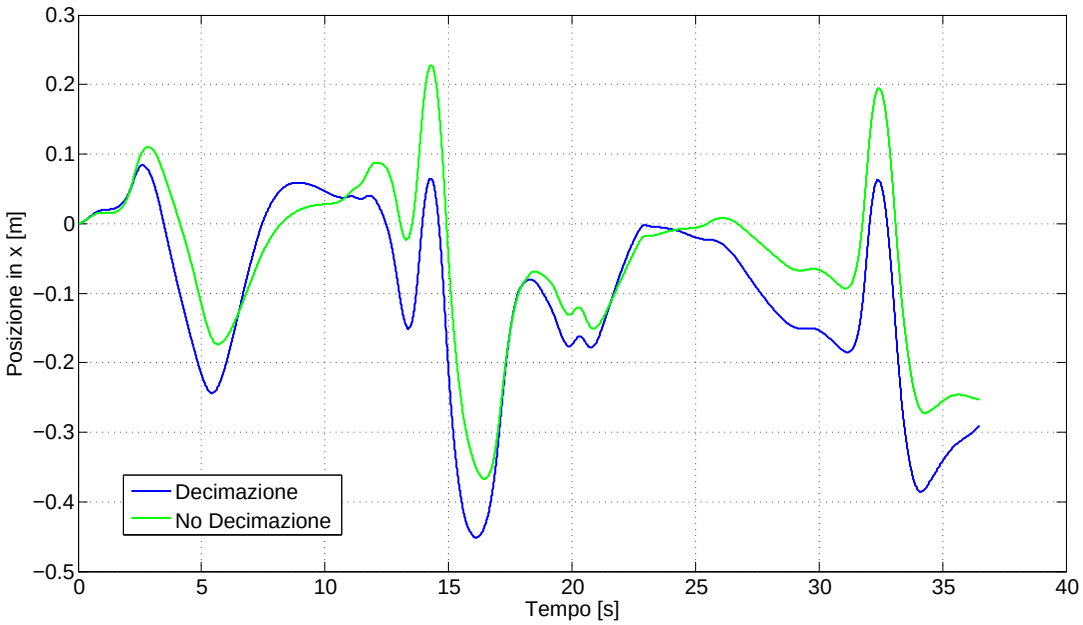


Figura 4.18: confronto della pos. long. con o senza decimazione, 2° Es.

4.3 Analisi senza pitch

L'ultima sezione di questo capitolo approfondisce una delle principali problematiche pratiche degli algoritmi di motion cueing: l'utilizzo del pitch. Come specificato più volte nei paragrafi precedenti, nella maggior parte delle prove effettuate si è sempre cercati di non eccedere nell'utilizzo di questo grado di libertà, in quanto il nostro scopo sarebbe principalmente quello di utilizzarlo per la tilt coordination e invece spostamenti repentini o posizioni angolari troppo marcate trasmettono ovviamente al pilota anche sensazioni di rotazione. Tali sensazioni, a cui non corrispondo adeguati stimoli visivi, producono la cosiddetta *motion sickness* che può essere tradotta con l'espressione *mal di simulatore*. Per questo motivo in molte applicazioni anche estramente professionali dei simulatori di guida (ad esempi in formula1) molti piloti quando devono stare ore al simulatore preferiscono bloccare l'utilizzo del pitch e accontentarsi delle sensazioni di guida trasmesse tramite i limitati spostamenti longitudinali che possono riprodurre solo le alte frequenze dei segnali di velocità e accelerazione (le basse frequenze porterebbe ad uno sfruttamento inadeguato della piattaforma). La maggior parte degli algoritmi di motion cueing oggi esistenti (principalmente i cosiddetti *washout filters*) sono abbastanza efficaci nel riprodurre le alte frequenza mentre lo sono molto meno nel riprodurre le basse data le difficoltà precedentemente descritte legate ad un corretto uso della tilt coordination. Come detto nell'introduzione questo lavoro di tesi si inserisce nell'ambito della realizzazione di un completo algoritmo di motion cueing basato su MPC, con la speranza che l'applicazione di un controllo predittivo posso migliorare le prestazioni complessive soprattutto quindi in termini di utilizzo della tilt coordination, permettendone una perfetta sintonia con gli spostamenti longitudinali. Tuttavia un'analisi ingegneristica appropriata deve sempre prevedere il caso peggiore e cioè la possibilità che anche con il sistema di controllo proposto si garantisca di trasmettere al pilota delle buone sensazioni di guida utilizzando solo gli spostamenti longitudinali, qualora il pitch fosse ridotto al minimo o addirittura assente. L'analisi che seguirà illustra come anche tramite MPC sia possibile riprodurre fedelmente le alte frequenze anche senza look ahead.

4.3.1 Filtraggio passa alto

Disabilitando l'utilizzo del pitch è impensabile di riprodurre un riferimento come quello di figura 4.3 utilizzato fin'ora. Per prima cosa quindi, stile motion cueing classico, è necessario eseguire un filtraggio passa alto dello stesso:

$$W_{HF}(s) = \frac{s\tau}{1 + s\tau} \tag{4.5}$$

con $\tau = 1/\omega_0 = 1/(2\pi f_0)$.
La scelta della frequenza di taglio f_0 del filtro è uno degli aspetti chiave: se troppo bassa il sistema di controllo del motion cueing non sarà efficace in quanto per cer-

care di riprodurre le basse frequenze non filtrate si avrà uno sfruttamento inadeguato della piattaforma; viceversa se troppo elevata si corre il rischio di trasmettere troppo poco al pilota in termini di sensazioni di guida non sfruttando al meglio la piattaforma. Data questa analisi si è verificato che una frequenza di taglio adeguata e abbastanza conservativa è $f_0 \cong 1Hz$, mentre poi si è cercato di forzare il sistema verso i propri limiti con una frequenza di taglio inferiore $f_0 = 0.5Hz$ per verificare che comunque si riescono ad ottenere delle prestazioni adeguate.

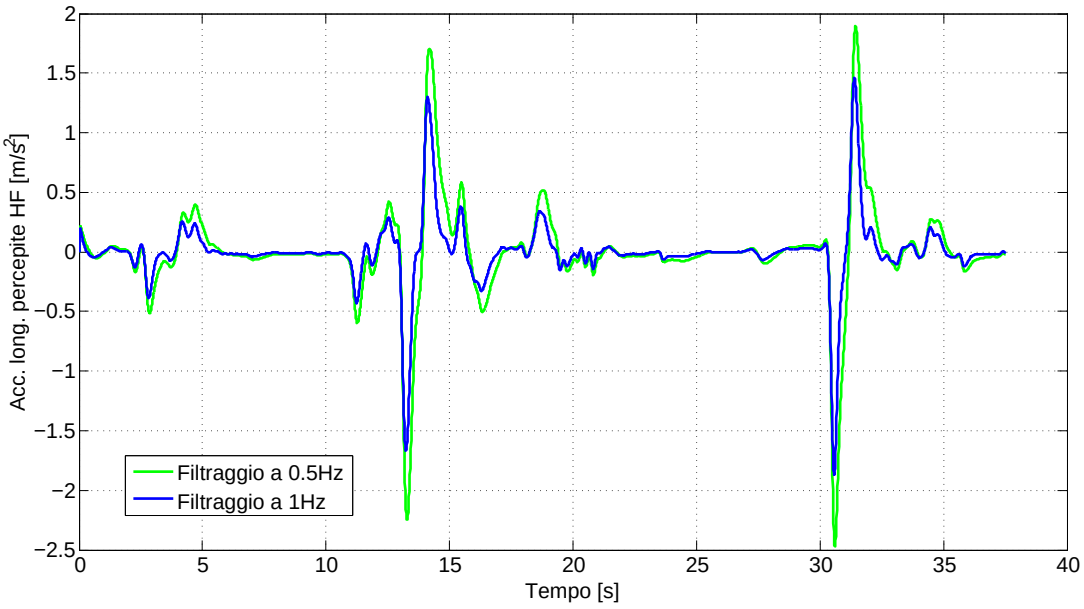


Figura 4.19: filtraggio passa alto del rif. in acc. long. percepita, $0.5Hz$ e $1Hz$

Frequenza 1 Hz

Sono di seguito riportati alcuni grafici relativi alle prestazioni ottenute con o senza look ahead. Si è cercato di scegliere il tipo di simulazione che più si addice alle due differenti strategie di generazione del riferimento, tenendo sempre in conto il fatto di mantenere i tempi di calcolo limitati. A tal proposito si è scelto di effettuare le prove senza look ahead con $T_{pred} = 0.5s$ senza decimazione della predizione, mentre quelle con look ahead (data l'importanza di avere un orizzonte di controllo elevato) sono state fatte con il seguente passo di decimazione:

$$T = \begin{bmatrix} 0.1 & 0.5 & 1.5 \end{bmatrix} \qquad K = \begin{bmatrix} 1 & 2 & 5 \end{bmatrix}$$

con un $N_{p,eq} = 50 = N_p$.
Osservando la figura 4.20 è possibile vedere come impostando una frequenza di taglio opportuna (ad esempio con i dati a disposizione $1Hz$) l'inseguimento del

riferimento è molto buono; inoltre a differenza del caso generale le differenze ottenute con o senza utilizzo di una predizione intelligente sono minime (si veda anche la tabella delle prestazioni). Significativi a tal proposito i due grafici che seguono 4.21, 4.22 in cui si nota però come l'inseguimento del jerk dell'accelerazione longitudinale percepita presenti un leggero ritardo nel caso senza look ahead, prevedibile in quanto la logica con cui si crea il riferimento (costante da $t+1$ a $t+N_p$ pari al valore in t) introduce inevitabilmente almeno un passo di ritardo. E' poi riportato in figura 4.23 lo sfruttamento longitudinale della piattaforma senza look ahead, del tutto simile a quello ottenuto con look ahead.

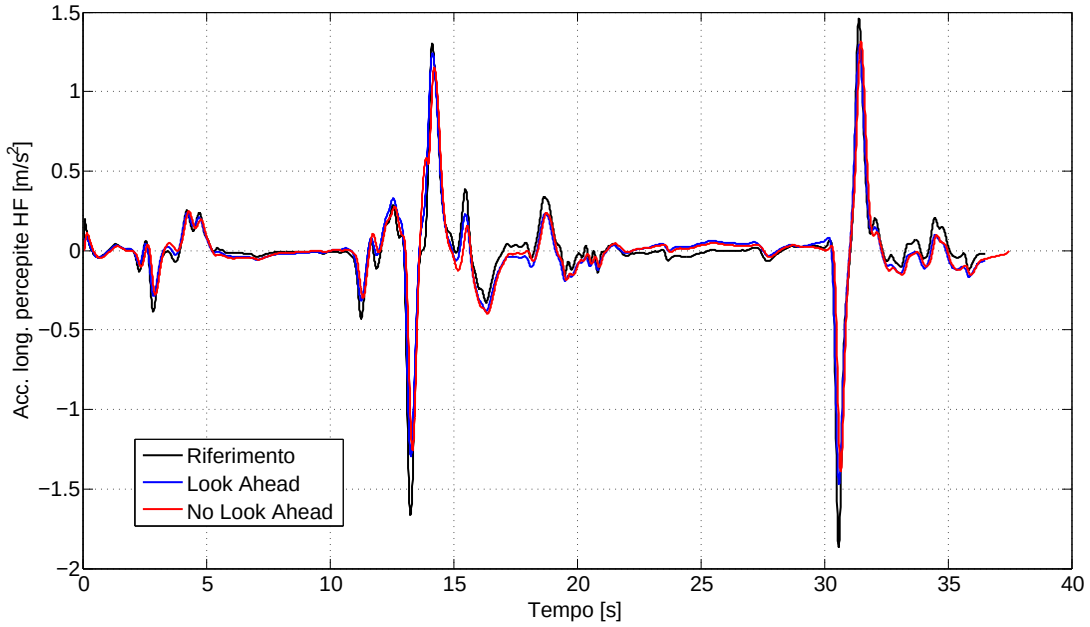


Figura 4.20: confronto tra le acc. percepite con o senza Look Ahead, $f_0 = 1Hz$

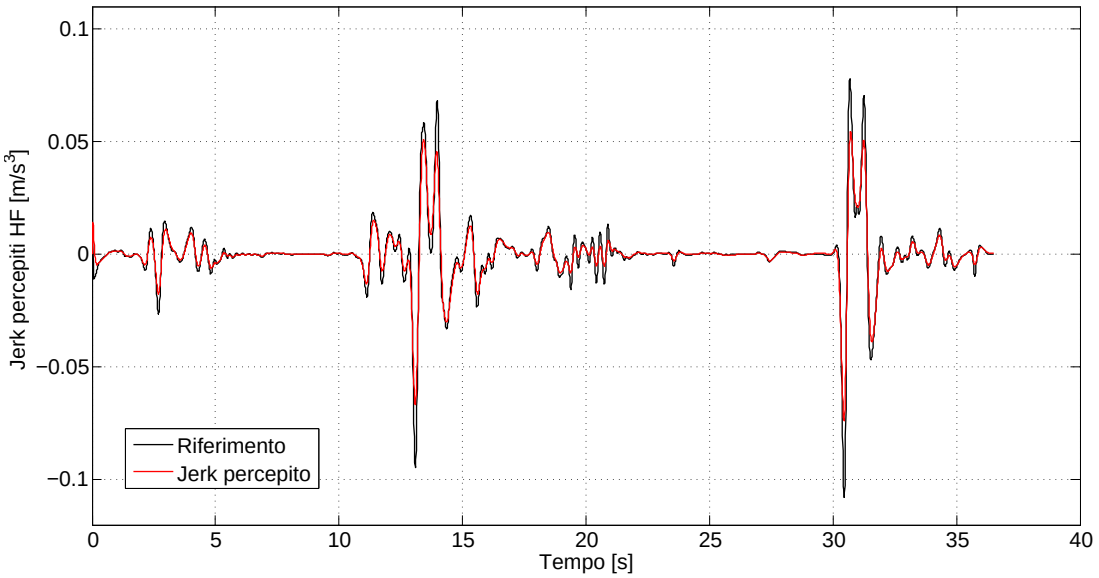


Figura 4.21: jerk percepito utilizzando Look Ahead, $f_0 = 1Hz$

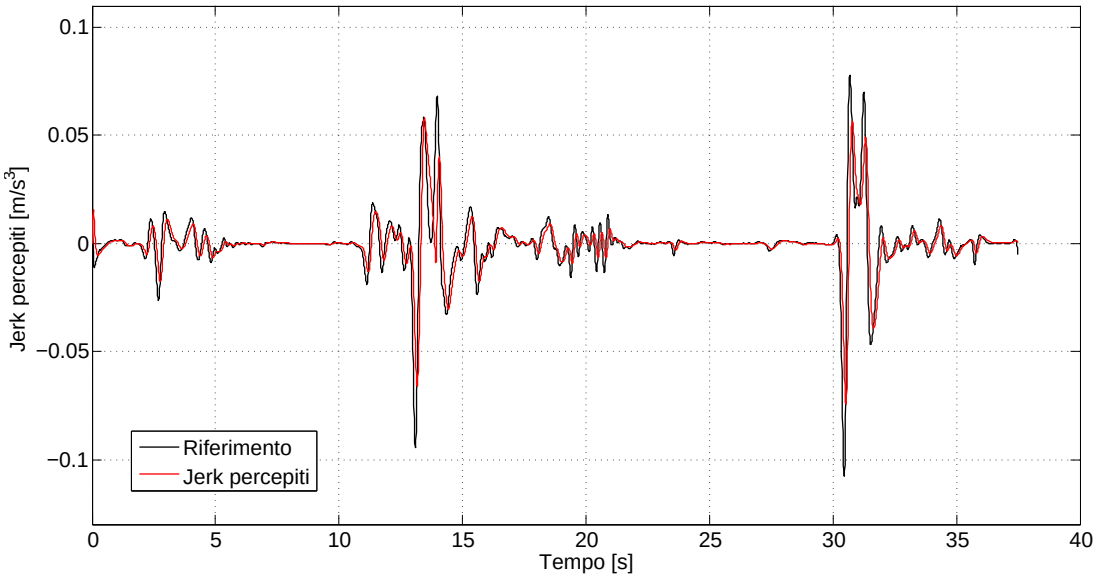


Figura 4.22: jerk percepito senza Look Ahead, $f_0 = 1Hz$

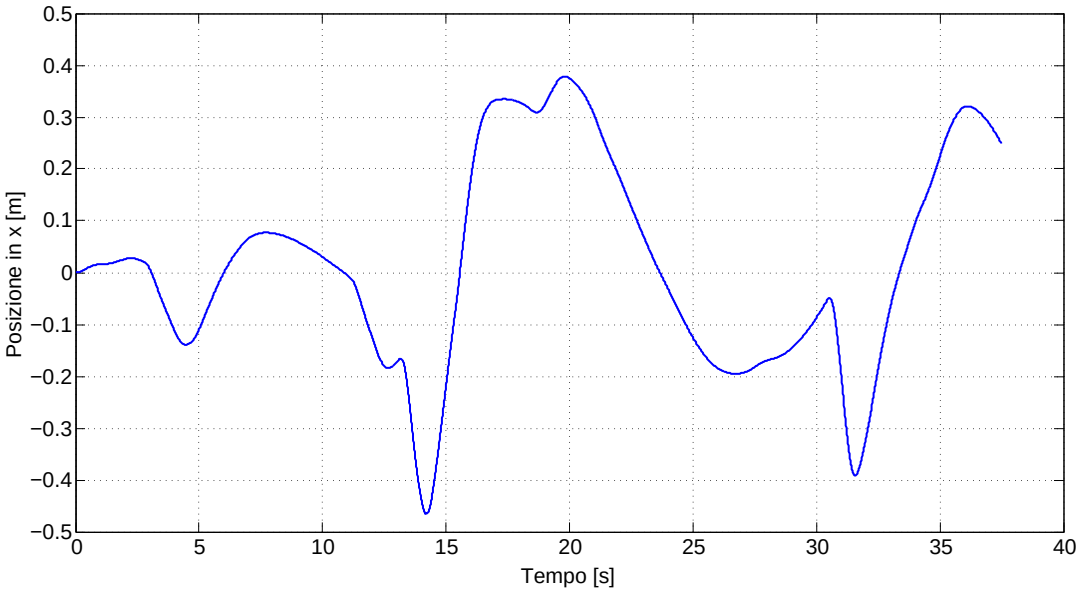


Figura 4.23: posizione longitudinale senza Look Ahead, $f_0 = 1Hz$

Frequenza 0.5 Hz

In questo secondo caso si è cercato di scegliere una frequenza di taglio meno conservativa per cercare di verificare i limiti del sistema di controllo nel caso in questione. Come prima si sono effettuate le prove con o senza look ahead, mantenendo la stessa scelta di T_{pred} e dei passi di decimazione. Anche in questo caso è possibile notare come le differenze con o senza look ahead non siano molto marcate (figura 4.24) e si traducano per lo più in un ritardo nel jerk percepito esattamente come prima (figure 4.25, 4.26). Lo sfruttamento della piattaforma senza look ahead (figura 4.27) è adeguato come nel caso precedente, tuttavia data una frequenza di taglio minore che lascia passare maggiormente le basse frequenze, si ha l'aspettato calo delle prestazioni in termini di inseguimento del riferimento come mostrato nella tabella.

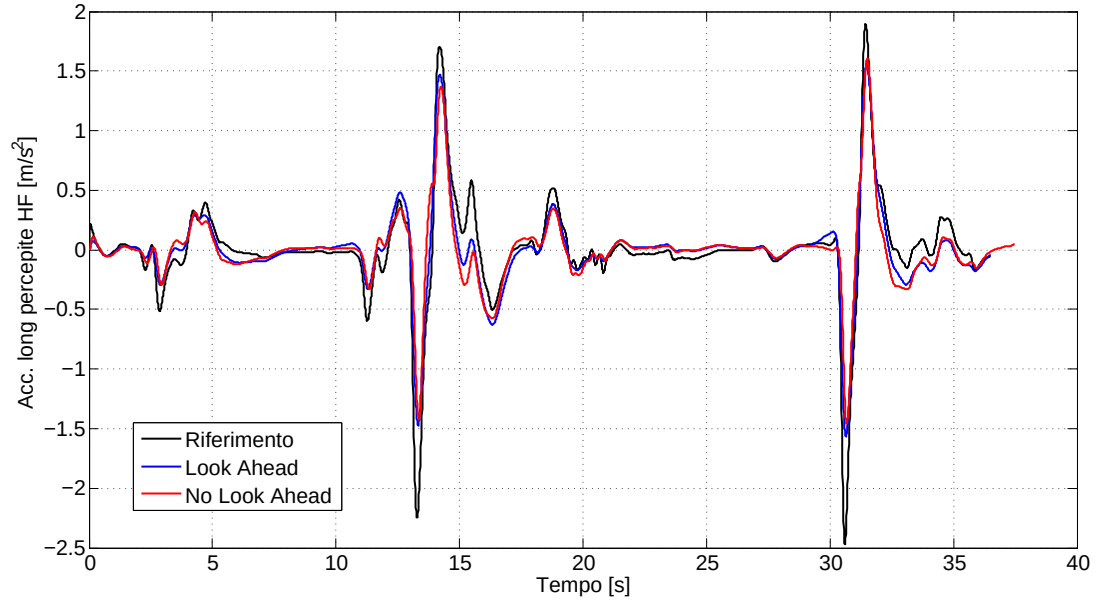


Figura 4.24: confronto tra le acc. percepite con o senza Look Ahead, $f_0 = 0.5Hz$

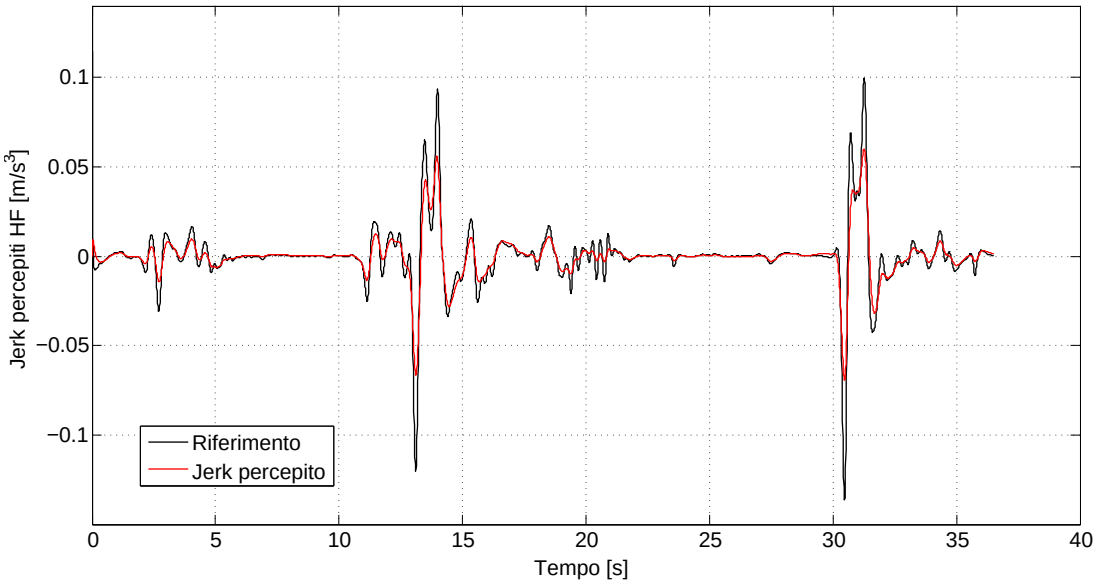


Figura 4.25: jerk percepito utilizzando Look Ahead, $f_0 = 0.5Hz$

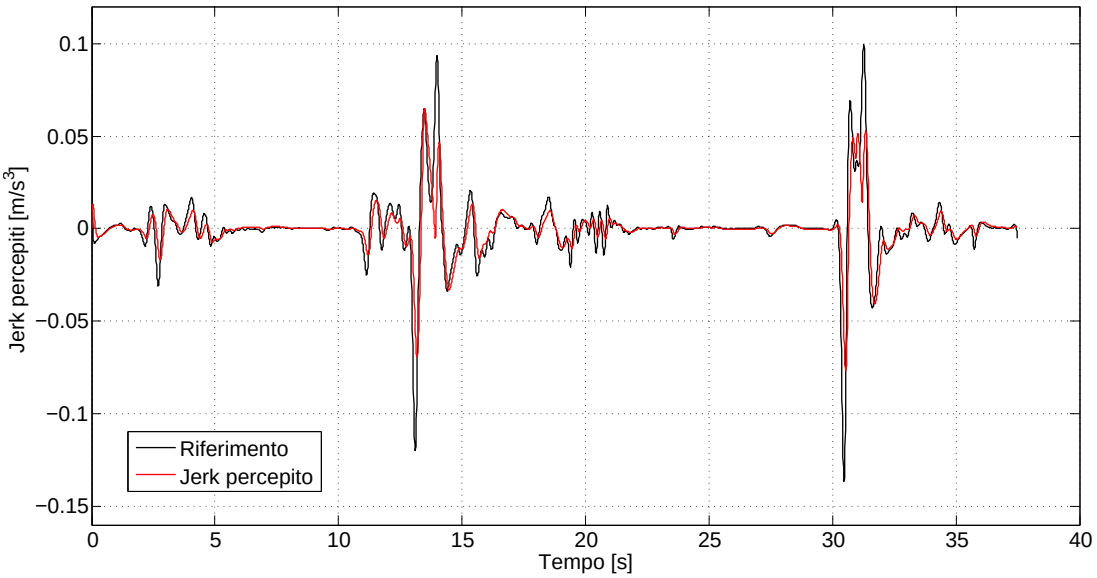


Figura 4.26: jerk percepito senza Look Ahead, $f_0 = 0.5Hz$

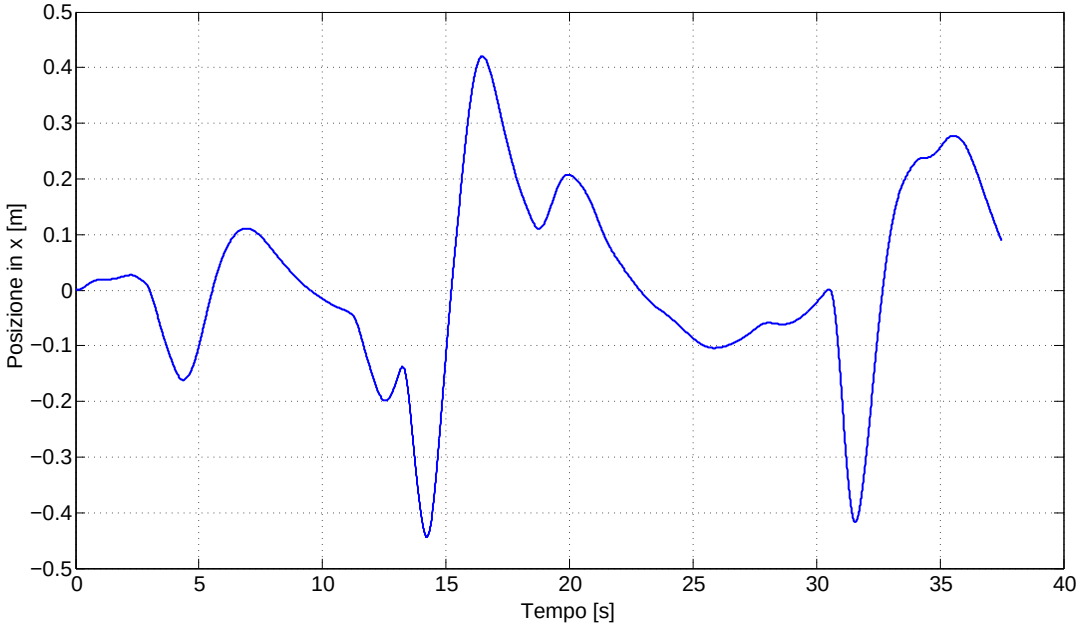


Figura 4.27: posizione longitudinale senza Look Ahead, $f_0 = 0.5Hz$

	Tempo di simulaz.	**Norma err.**	**Freq. di taglio**
Look Ahead	30s	4.08	1Hz
No Look Ahead	31s	7.05	1Hz
Look Ahead	30s	9.07	0.5Hz
No Look Ahead	31s	12.62	0.5Hz

Tabella 4.6: prestazioni senza utilizzare pitch

L'analisi effettuata è complessivamente molto interessante in quanto dimostra come anche in assenza di pitch e di un riferimento per la predizione intelligente, sia possibile inseguire le alte frequenze del segnale di riferimento in termini di accelerazione longitudinale percepita. E' molto importante garantire un comportamento adeguato del motion cueing in questo contesto, in quanto esso può essere interpretato come il *worst case* in cui cioè senza aiuto della tilt coordination e dei vantaggi propri di un controllo predittivo, sfruttando solo un'ottimizzazione vincolata basata su di un modello percettivo, si riesce comunque a garantire un minimo standard di qualità nelle prestazioni del motion cueing. Inoltre questo tipo di test ci ha permesso di capire l'importanza del pitch e di un utilizzo combinato pitch-longitudinale, per cercare nel caso generale senza filtraggio di ottimizzare l'inseguimento del riferimento. In questo contesto era emersa chiaramente l'importanza della predizione e il fatto che quindi il controllo con look ahead, dato un orizzonte di predizione adeguato, fosse indispensabile per fornire delle ottime prestazioni. Infatti (come è stato approfondito nella tesi del collega Mauro P. [2]) il sistema spontaneamente sfrutta il pitch per riprodurre per lo più le basse frequenze e aiutare il longitudinale nelle frenate più intense, e questo tipo di accoppiamento si può realizzare in modo efficace solo disponendo di una predizione adeguata.

Capitolo 5

Generazione real time del riferimento

Come più volte accennato nel corso di questa tesi ed in particolare nel precedente capitolo, lo scopo del lavoro svolto è progettare un algoritmo di motion cueing fisicamente realizzabile in vista di un'implementazione reale sul simulatore VI-grade. A tal proposito si pone il problema della scelta del riferimento: le soluzioni illustrate nel precedente capitolo 4 presentano entrambe dei limiti dal punto di vista realizzativo. Il look-ahead infatti non è fisicamente realizzabile, in quanto presuppone di disporre del futuro del giro che si sta attualmente compiendo, il che è ovviamente assurdo, tuttavia si è visto che fornisce prestazioni molto buone. D'altro canto l'assenza di look-ahead, situazione invece realistica, consente inseguimenti discreti ma viene a mancare la possibilità di sfruttare appieno le potenzialità dell'MPC e si intuisce che si potrebbe comunque fare di meglio, specialmente nella condizione di guida su pista in cui il tracciato è noto ed una qualche forma di previsione intelligente può essere fatta. Queste considerazioni hanno portato al concetto di matching o generazione online del riferimento: tale idea, sviluppata e realizzata, può essere considerata a tutti gli effetti una valida alternativa da utilizzare online a quelle precedentemente descritte.

5.1 Matching: idea e strategia

Come già accennato nell'introduzione, l'idea alla base del matching è che, nelle condizioni di guida su pista, il tracciato è noto in anticipo e si hanno a disposizione i dati telemetrici relativi al giro precedente, che vengono man mano aggiornati. Si esegue dunque l'MPC utilizzando come riferimento una versione "adattata" del giro precedente: nel caso di piloti semiprofessionisti si può assumere infatti che le variazioni da un giro all'altro siano di entità non troppo rilevante e dunque i giri passati possono essere utilizzati come riferimento, purché come detto opportunamente adattati. Nel caso di piloti professionisti, i quali riescono ad inanellare una

lunga serie di tornate con differenze marginali tra una e l'altra, tale adattamento sarà minore e l'idea di base si dimostra ulteriormente valida. L'adattamento di cui si è accennato finora consiste in semplici modifiche apportate online alla telemetria utilizzata come riferimento: dilatazioni, scalature e traslazioni. Queste modifiche, opportunamente combinate, danno luogo ad una serie di possibili telemetrie, che vanno confrontate con una finestra temporale della telemetria attuale, ad esempio gli ultimi 5 s. In tal modo si sceglie l'adattamento che meglio approssima l'andamento attuale ed esso viene usato per andare a fare predizione: il riferimento terrà dunque conto della conformazione della pista e dello stile di guida del pilota, assumendo che esso non vari considerevolmente da un giro all'altro. Qualora ciò accadesse, ovvero quando nessuna delle possibili generazioni del riferimento si adatti bene ai dati attuali, secondo un indice di qualità impostato, tale strategia non agisce e si utilizza al suo posto quella con riferimento costante, realistica ma peggiore nelle performance. Complessivamente si è dunque in grado di operare real time e di fare predizione intelligente: i tempi di calcolo si prevede siano leggermente maggiori del caso con look-ahead a causa dei riferimenti da generare online ad ogni passo, tuttavia sono stati presi opportuni accorgimenti per ridurre l'onerosità computazionale.

5.2 Simulazioni

In figura 5.1 si presenta una prima prova di utilizzo del matching, confrontandola con gli andamenti ottenuti con pesi identici ma che si differenziano rispettivamente per la presenza e l'assenza del look-ahead. Come si può vedere, l'andamento è tendenzialmente simile a quello con look-ahead, segno che utilizzare la telemetria passata per generare online il riferimento è una buona strategia, anche se in certi casi, ad esempio nell'accelerazione seguente alla seconda frenata impegnativa, il comportamento ricalca esattamente quello con riferimento costante (constant reference, CR nei grafici). L'errore complessivo, riportato per questa e per le successive simulazioni in tabella 5.1, è una via di mezzo dei due; i tempi di calcolo della simulazione sono come predetto leggermente maggiori ma comunque dell'ordine della durata della simulazione riprodotta. E' possibile osservare inoltre che gli ingressi presentano delle variazioni piuttosto brusche, che comportano dei picchi indesiderati nel jerk. Questo fatto è di facile spiegazione: applicando il controllo con generazione online del riferimento, che quindi potenzialmente cambia segnale ad ogni passo ed inoltre alternando il suo uso a quello con look-ahead, nonostante sia stata usata l'accortezza di rendere continuo il segnale di accelerazione percepita, si hanno comunque delle discontinuità nella sua derivata (osservare a titolo di esempio il tratto compreso tra 3.4 s e 3.8 s messo in evidenza in figura 5.2) che si traducono nei picchi presenti nel jerk. Gli ingressi hanno andamento molto variabile proprio a causa dei continui cambiamenti di riferimento e della necessità di rendere quest'ultimo continuo. Si deve cercare dunque, per quanto possibile, di limitare

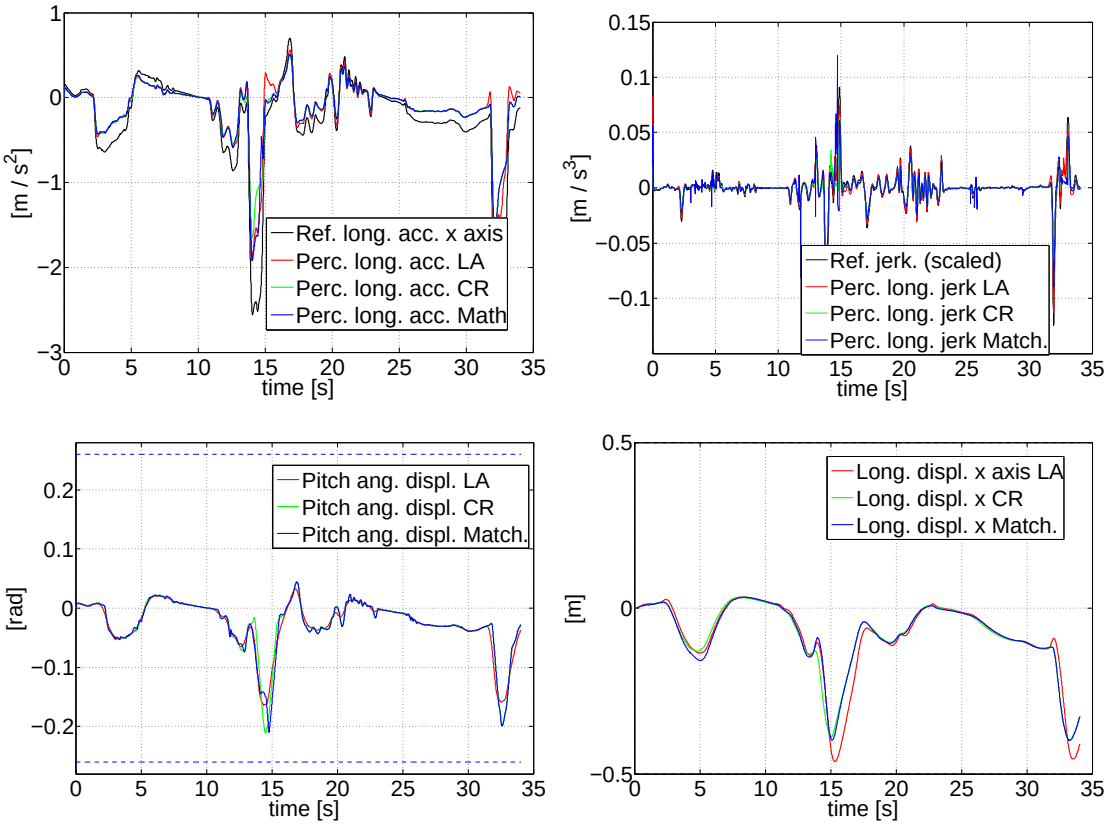


Figura 5.1: Simulazione di matching, confronto con LA (look-ahead) e CR (riferimento costante).

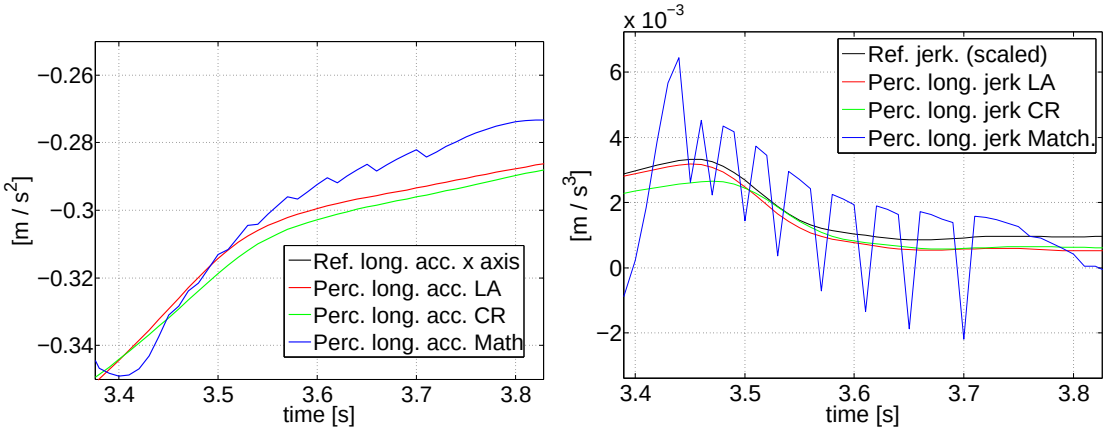


Figura 5.2: Dettaglio dell'accelerazione percepita e del jerk per la simulazione di figura 5.1.

tali discontinuità agendo sia sui pesi del controllo predittivo sia sui parametri della funzione di matching, come fatto ad esempio nella simulazione mostrata in figura

Tabella 5.1: Parametri e pesi adottati per le simulazioni con matching.

	Fig. 5.1	Fig. 5.3	Fig. 5.4	Fig. 5.5	Fig. 5.6
Or. pred. [s]	0.5	0.5	1.5	1	1.5
T_i [s]	-	-	[0.1 0.5]	[0.1 0.5]	[0.4 1]
k_i	-	-	[1 2 5]	[1 2 5]	[2 3 5]
N. camp.	50	50	50	40	50
y_1	0	0	0	0	0
y_2	0.06	0.1	0.12	0.1	0.5
y_3	2	2	2.8	4	5
y_4	0.3	0.3	0.35	0.3	0.1
y_5	0.1	0.1	0.12	0.2	0.1
y_6	0	0	0	0	0
u_1	10^{-4}	10^{-5}	10^{-4}	10^{-5}	10^{-3}
u_2	10^{-5}	10^{-3}	10^{-5}	10^{-2}	10^{0}
Δu_1	10^{-5}	10^{0}	10^{0}	10^{0}	$3 * 10^{0}$
Δu_2	10^{-5}	10^{0}	10^{0}	10^{0}	10^{2}
Norma err.	16.1021	15.0952	11.0880	14.4104	10.8231
Durata [s]	36.1566	32.8291	30.0216	28.7739	36.3849

5.3: l'errore è leggermente inferiore al caso precedente grazie anche alle modifiche apportate agli ingressi che presentano ora variazioni meno brusche e di conseguenza il jerk fa altrettanto, anche se restano alcuni picchi di difficile cancellazione.

Decimazione Alla strategia di matching in uso si aggiunge ora la decimazione, già illustrata e commentata approfonditamente in sezione 4.2.2. Ci si aspetta di migliorare le performance dell'inseguimento a parità (circa) di tempi di calcolo, grazie al fatto di osservare una porzione maggiore di riferimento. Anche in questo caso si inizia mostrando il confronto con la tecnica look-ahead, per mettere in maggior evidenza le potenzialità della strategia qui introdotta. Come si può vedere da figura 5.4 l'inseguimento è molto buono nonostante il secondo ingresso, presenti delle variazioni brusche che però non si ripercuotono più di tanto sul jerk, a parte un unico vistoso picco. Lo sfruttamento della piattaforma per le due tecniche è abbastanza analogo, segno ulteriore della correttezza della generazione dei riferimenti online. L'errore è minore dei due casi precedenti grazie alla decimazione mentre il tempo di calcolo è quasi lo stesso (si veda sempre tabella 5.1). Siccome è stata messa in debita rilevanza l'importanza di non fornire sensazioni contraddittorie, in termini di jerk, si è cercato di modificare i parametri in gioco per eliminare del tutto i picchi indesiderati, cosa che come detto non risulta affatto facile vista la natura della tipologia del controllo adottato, che ne favorisce spontaneamente la nascita. Si è riusciti tuttavia ad arrivare alla simulazione di figura 5.5, nella quale non sono presenti picchi fortemente anomali nel jerk. Quest'ultimo, se osservato nel dettaglio, non si dimostra proprio perfetto nell'inseguimento, a causa del fatto

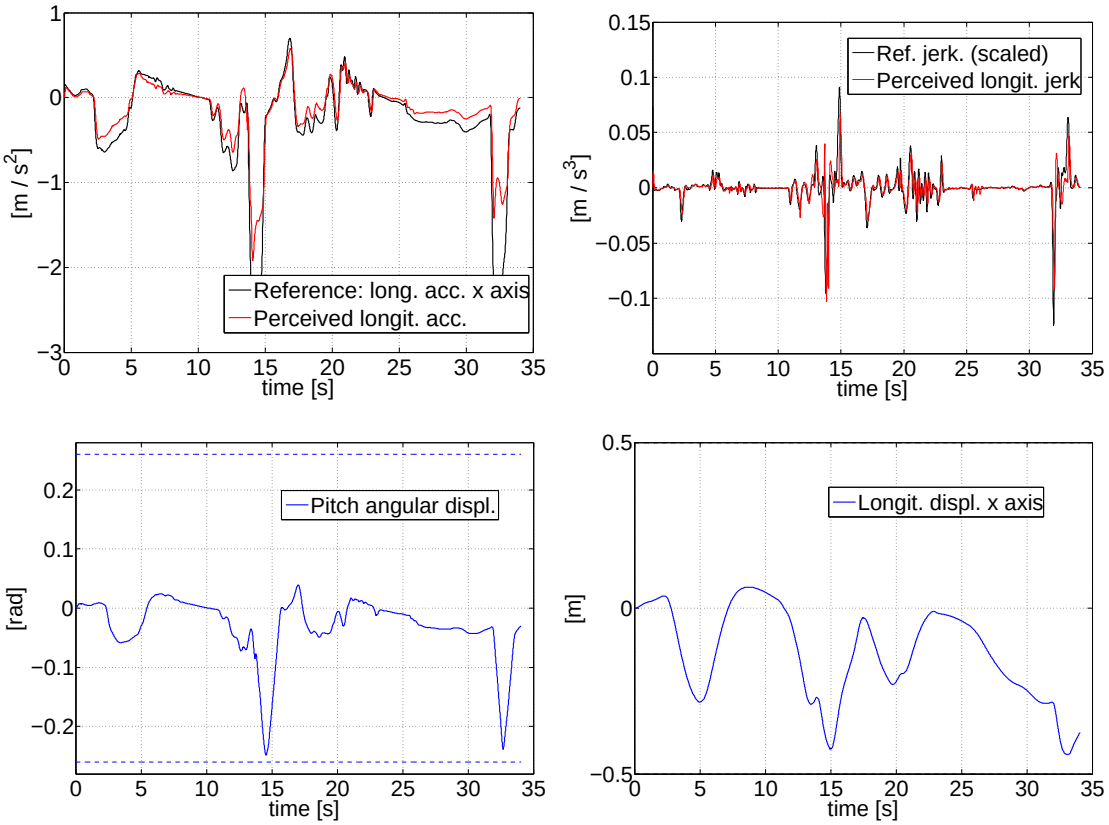


Figura 5.3: Simulazione di matching con discontinuità ridotte.

che le modifiche apportate hanno condotto ad una prestazione più conservativa e dunque ad un errore maggiore in accelerazione percepita, tuttavia l'andamento può essere considerato abbastanza soddisfacente.
Finora sono state mostrate alcune prove con decimazione con tre intervalli di campionamento differenti, scelti come nel capitolo precedente, dal momento che si era mostrato come tale soluzione portasse a risultati molto buoni. Tuttavia, come è lecito attendersi, le cose vanno abbastanza bene anche variando la scelta degli intervalli e di come ripartire i campioni, purché la cosa sia fatta con criterio, come mostrato in figura 5.6: l'inseguimento è abbastanza buono fatta eccezione per la seconda frenata, che presenta delle anomalie nella fase conclusiva, che portano a dei picchi nel jerk. Gli ingressi (in figura 5.6) non sono però particolarmente bruschi nelle loro variazioni, come è confermato dal fatto che il resto del riferimento è inseguito piuttosto bene, inclusa la prima frenata.
In conclusione, la strategia di generazione del riferimento adottata dovrebbe portare in teoria ad un buon inseguimento, pari come prestazioni alla strategia con look-ahead, o di poco inferiore. In realtà gli ingressi, per come sono costruiti, portano a delle discontinuità nel jerk, pur a fronte di un buon inseguimento di accelerazione. Tali discontinuità sono intrinseche alla natura del cueing così concepito e dunque

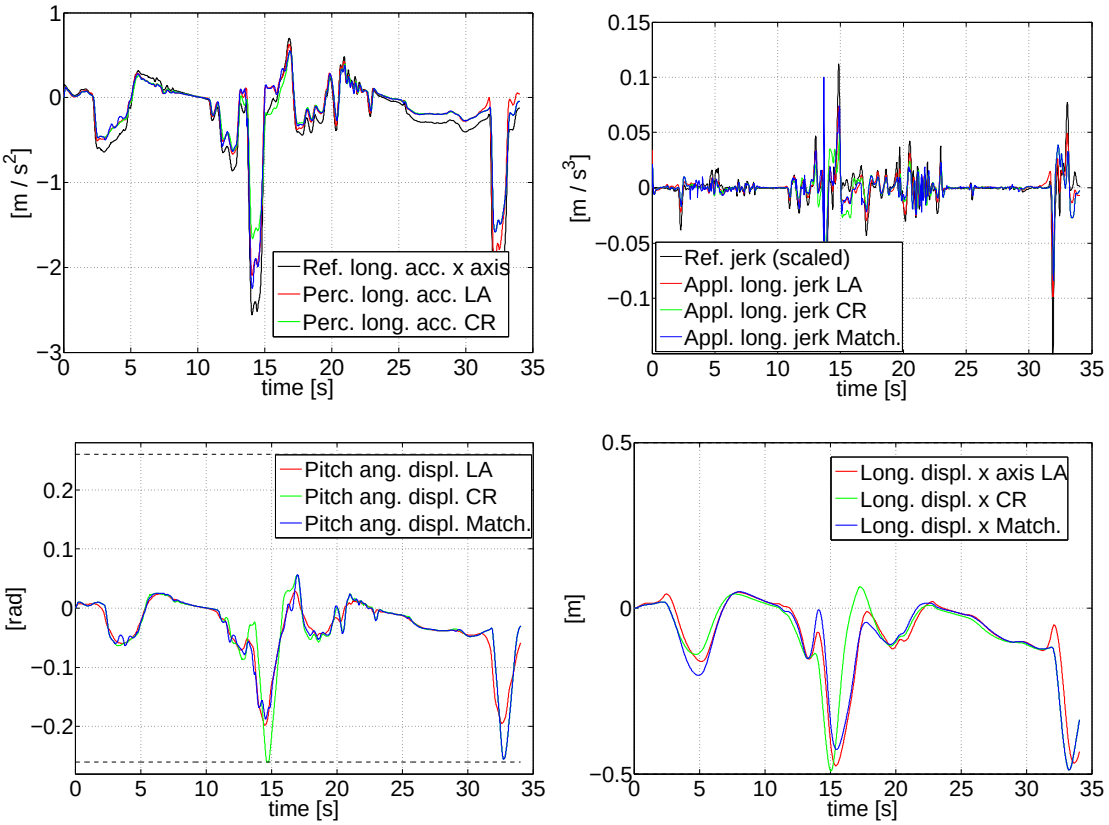


Figura 5.4: Simulazione di matching con decimazione, confronto look-ahead (LA) ed andamento corrispondente con riferimento costante (CR).

più che eliminarle si può cercare di ridurne il numero e l'ampiezza adottando ingressi più dolci, che tendenzialmente comportano però un peggioramento delle prestazioni del controllore. Ciononostante in termini unicamente di accelerazione percepita la strategia adottata si è rivelata molto valida anche nei test. Essa non presenta alcun problema ad essere integrata con la decimazione illustrata nel capitolo precedente e dunque complessivamente si pone come una buona soluzione per il problema concreto, real time, di motion cueing che VI-grade sta affrontando. Non sono da escludere migliorie all'algoritmo che "addolciscano" le discontinuità portando ad andamenti nel jerk meno fuorvianti, come pure la possibilità di andare a generare riferimenti direttamente nel jerk, in modo che le derivate successive, seppur discontinue, non creino problemi. Tutti questi sono però sviluppi futuri destinati ad essere indagati da coloro i quali proseguiranno il presente lavoro.

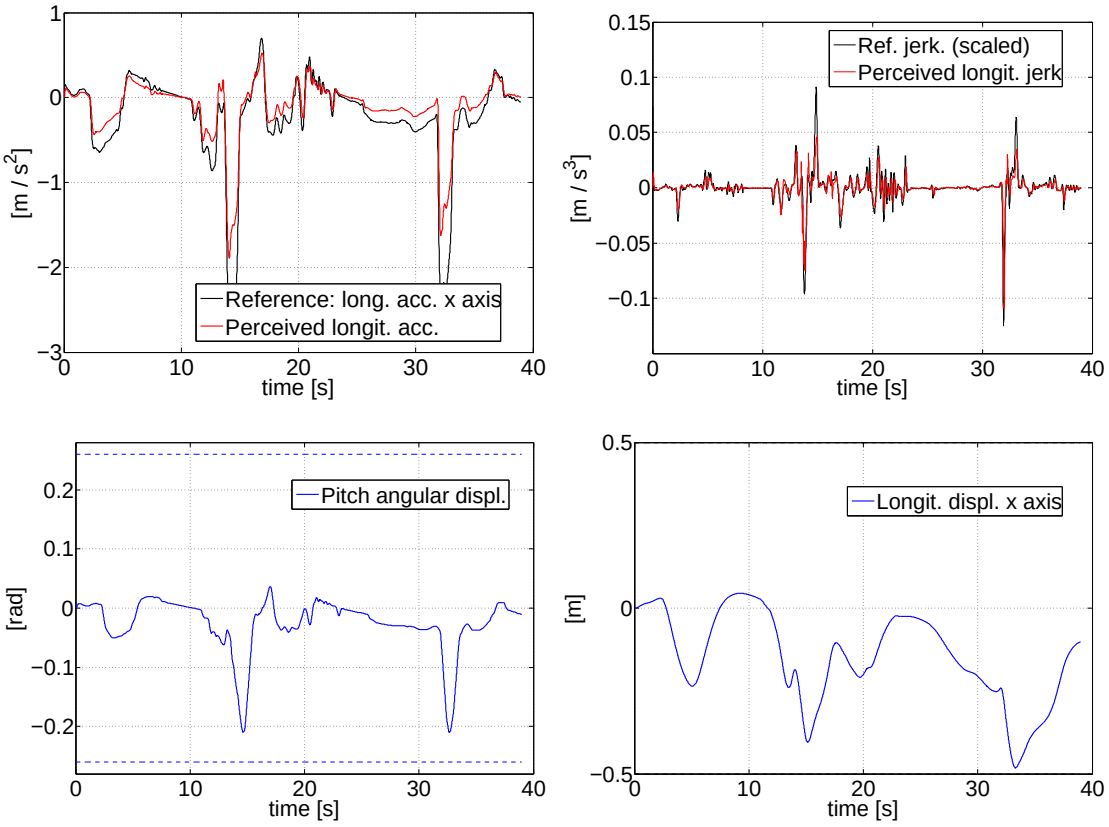


Figura 5.5: Simulazione di matching con discontinuità ridotte e decimazione.

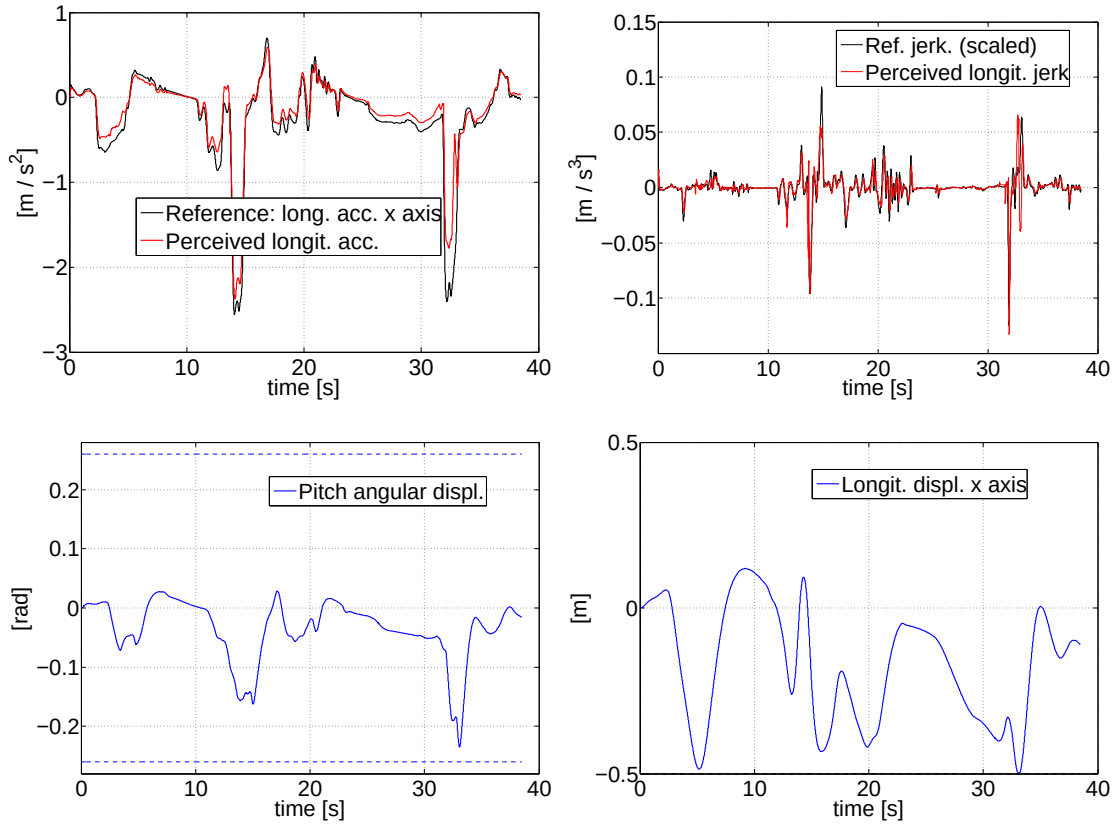


Figura 5.6: Simulazione di matching con decimazione con intervalli e campioni diversi.

Capitolo 6

Conclusioni e Sviluppi futuri

In questo lavoro di tesi si è affrontato il problema della generazione dei riferimenti per un algoritmo di motion cueing basato su MPC. Si è proposto un metodo innovativo basato sull'utilizzo di una telemetria di riferimento e, pur realizzandolo con un'implementazione di base particolarmente semplice, si è dimostrato come sia verosimile un utilizzo real time dello stesso. Se ne è verificata l'efficacia confrontandolo con le prestazioni ottenibili con la strategia con *Look Ahead*, in cui si utilizza il futuro del segnale stesso per fare predizione. Si è affrontata poi la problematica dei tempi di calcolo richiesti dal MPC adottando una modifica dell'algoritmo standard che si basa sulla decimazione dell'orizzonte di predizione, dimostrando come a un piccolo calo delle prestazioni in termini di inseguimento del riferimento corrisponda però una dimuzione significativa dei tempi di calcolo. Le prove sono state effettuate sull'accoppiamento *pitch-longitudinale* illustrando come, ad una corretta taratura del MPC, corrispondano ottime prestazioni in termini di tempo di calcolo e di inseguimento del riferimento, con un ampio utilizzo delle risorse della piattaforma. Nonostante i buoni risultati ottenuti con questo lavoro di tesi, è ancora ampio il lavoro da fare per implementare sul simulatore reale l'algoritmo di motion cueing proposto. Di seguito sono riportate le principali problematiche ancora aperte, su cui sarà necessaria un'analisi approfondita:

1. Analisi dei sottosistemi Σ_1, Σ_2, Σ_4 (3.2)

2. Indagine accurata dei tempi di calcolo dell'ottimizzatore

3. Estensione del metodo di generazione del riferimento proposto

4. Confronto dei segnali generati dal motion cueing proposto con quello già presente nel simulatore

Tutte queste quattro questioni sono molto importanti e dovranno essere affrontate prima di pensare di portare in piattaforma l'approccio proposto. Verosimilmente dovranno anche essere affrontate nell'ordine esposto: innanzitutto sarà importante

trasportare le analisi effettuate anche in termini di accoppiamento *roll-laterale* (sottosistema Σ_4) e di spostamenti e rotazioni lungo l'asse z (Σ_2, Σ_1), verificando un corretto funzionamento del motion cueing anche per questi gradi di libertà.
Successivamente sarà indispensabile eseguire anche un'indagine accurata dei tempi di calcolo richiesti dall'ottimizzatore; infatti in tutte le simulazioni effettuate si è riportato solo il tempo complessivo di durata della stessa. Verificare che esso è inferiore ai secondi di giro simulato non è sufficiente a garantire che sarebbe stato possibile agire real time: è necessario vericare che ogni passo dell'ottimizzazione è avvenuto in meno di $0.01s$ (che rappresenta il periodo fondamentale di lavoro T_s) affinchè ciò sia possibile.

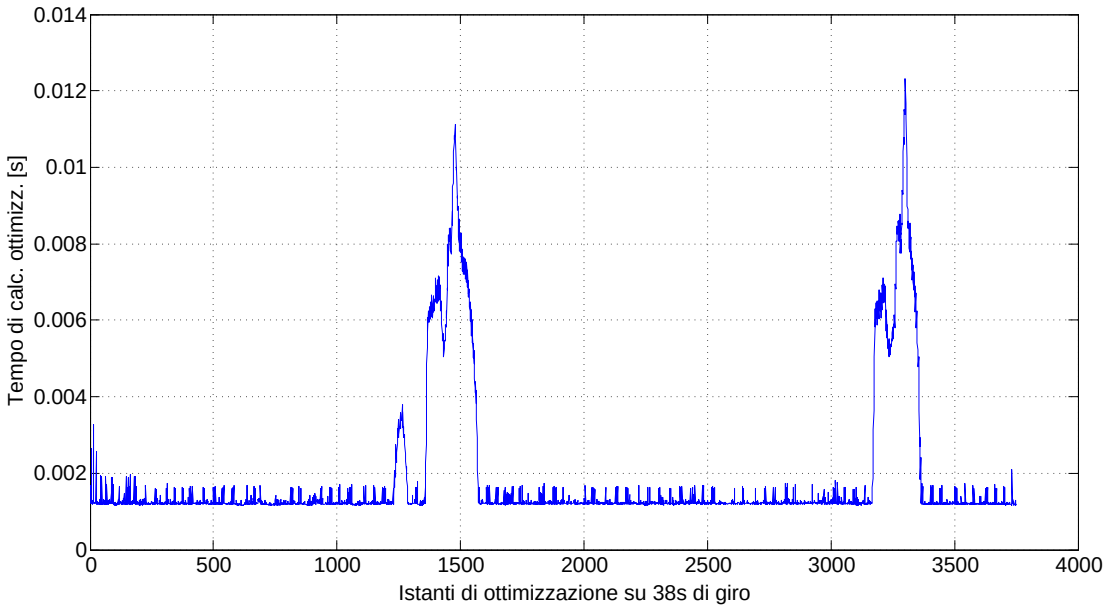


Figura 6.1: Tempi di calcolo dell'ottimizzatore ad ogni passo

La figura 6.1 mostra i risultati di una tipica simulazione di quelle riportate nel capitolo precedente: la durata della simulazione complessiva ($30s$) è inferiore ai secondi di giro simulati ($38s$) tuttavia in alcuni istanti il tempo richiesto dalla sola ottimizzazione (esclusi quindi i tempi di MATLAB che con un'implementazione in un linguaggio più efficiente possono essere considerati quasi annullati) è stato maggiore del periodo T_s fondamentale di $0.01s$. Ciò è accaduto in corrispondenza dei punti critici del segnale (nelle due frenate) ed è legato al fatto che l'ottimizzazione in quegli istanti è molto vicina ai vincoli del sistema e questo ne rallenta le operazioni. Si noti come comunque anche negli istanti di tempo più critici non si ecceda di molto $0.01s$: in altre prove effettuate è capitato di raggiungere al più $0.02s, 0.03s$ ma con una corretta taratura del MPC e uso della decimazione, si ha avuto l'impressione che non dovrebbe essere difficile già solo con questi accorgimenti mantenere il tempo di calcolo di un passo dell'ottimizzatore entro i limiti

richiesti; tuttavia sarà indispensabile un'indagine accurata a proposito.
Anche il metodo proposto di generazione del riferimento può essere esteso e modificato: infatti, nonostante già con la semplice implementazione proposta l'aumento dei tempi di calcolo sia contenuto, è auspicabile un'implementazione più efficiente dello stesso, basata però sempre sull'idea proposta che si è dimostrata appropriata per il problema in esame. E' possibile poi pensare di trasportare in coordinate spaziali l'algoritmo di matching proposto, in quanto intuitivamente sembra essere più efficace e robusto disporre di un riferimento ideale in termini di coordinate spaziali (posizione cioè nel tracciato) piuttosto che temporali. Inoltre questa prima versione di matching permette di generare il riferimento solo in termini di accelerazione percepita: si potrebbe pensare ad una procedura simile per cercare di generare anche un riferimento in termini di posizione longitudinale della piattaforma.
Infine un ulteriore passo fondamentale prima di trasportare in piattaforma il motion cueing proposto, sarà quello di confrontare i segnali di accelerazione e velocità prodotti dal nostro algoritmo con quelli ottenuti dal motion cueing già presente in piattaforma; soprattutto sarà indispensabile eseguire un confronto accurato nei segnali di controllo in posizione della piattaforma e di pitch prodotti nei due casi. Inoltre, pur essendo la logica del motion cueing proposto basato su MPC molto differente da quella del motion cueing già presente, è indispensabile capire bene dove il nostro algoritmo funziona in modo diverso e quali vantaggi o svantaggi può dare in termini di percezione questa differenza. Infatti non va dimenticato che il motion cueing già presente funziona complessivamente abbastanza bene e rappresenta quindi sicuramente un ottimo elemento di confronto per validare il nostro metodo che ambiziosamente si pone di cercare di migliorare la prestazioni già garantite da quello presente.

Appendice A

Script MATLAB per l'MPC

In questa appendice si introducono e si descrivono per sommi capi i file MATLAB utilizzati per eseguire MPC, sia col controllore implementato in questa tesi sia utilizzando il toolbox di Bemporad, al fine di facilitare il lavoro futuro di coloro i quali succederanno all'autore nello sviluppo del progetto per VI-grade.

A.1 Eseguire MPC con Bemporad

Come già spiegato nel capitolo 1, configurare i parametri del toolbox di Bemporad non è così immmediato; inoltre per gli scopi di questa tesi non ne sono state sfruttate appieno tutte le potenzialità, per le quali si rimanda alla userguide [14]. Lo script utilizzato in questa tesi è la funzione *mpcbempolim.m*. La chiamata della funzione prevede la seguente sintassi:

```
[y_bemp,u_bemp]=mpcbempolim(Ap,Bp,Cp,Dp,N_sim,Ts,Nc,Np,rumore,...
...bRif_Cost,bPasso,bSoglia,rif,CIs,CIu,rw_du,rw_u,rw_out,ylim,ulim)
```

In particolare, essa restituisce in uscita l'ingresso calcolato tramite MPC e le uscite del sistema in analisi calcolate con tale ingresso lungo tutta la durata della simulazione; per quanto riguarda gli ingressi, essi sono:

- le matrici del sistema del sistema discreto accoppiato su cui si desidera fare MPC, ovvero 2.8, cioè la serie dei sistemi meccanico e vestibolare ma, si badi bene, non si tratta di quello con riscrittura alternativa dello stato;
- i passi di simulazione N_{sim}, il tempo di campionamento T_s ed i passi di controllo e predizione N_c e N_p;
- la varianza del rumore di ingresso (che nel corso di questa tesi era considerato assente);
- una serie di valori booleani che indicano se effettuare o meno look-ahead, se prendere il riferimento al passo attuale o meno e l'eventuale presenza della soglia di percezione (da utilizzare impostando i limiti);

- il segnale di riferimento ed i vettori contenenti le condizioni iniziali, i pesi ed i limiti relativi ad ingressi ed uscite.

Il file crea un oggetto di tipo *mpc* e le matrici di pesi e limiti e le assegna a tale oggetto. Si decide di non utilizzare i soft constraints ponendo $ECR = 0$, si assume assenza di disturbi di ingresso ed uscita e si esegue la simulazione.

A.2 Eseguire MPC con le funzioni realizzate

Per eseguire il controllo tramite la funzione sviluppata nel corso di questa tesi e seguendo quanto riportato in sezione 2.3, è necessario chiamare la funzione *mpclim.m*, la quale presenta una sintassi molto simile a quella da utilizzare con Bemporad:

```
[y u] = mpcclim(A,B,C,N_sim,Np,Nc,rif_cost,rif,CI_x,CI_u,...
...pesi_du,pesi_u,pesi_out,y_lim,u_lim,range)
```

Essendo i parametri praticamente gli stessi del caso precedente, non se ne procede alla descrizione e si passa a descrivere quanto svolto dalla funzione. Essa innanzitutto chiama la funzione ausiliaria *mpcgain.m* con la seguente sintassi:

```
[Phi,F,A_tot,B_tot,C_tot]=mpcgain(A,B,C,Np,Nc)
```

Scopo di tale funzione è costruire il modello con stato aumentato 2.10 per avere in ingresso Δu; essa calcola anche le uscite predette e costruisce le matrici 2.14. A questo punto la funzione principale, *mpclim.m*, imposta le condizioni iniziali del problema e calcola tutto ciò che è fattibile fare offline, ovvero le matrici dei pesi e quelle tempo-invarianti della programmazione quadratica, vale a dire A (2.29) e H (2.22). A questo punto inizia il ciclo che effettua la simulazione vera e propria, della durata desiderata, aggiornando ad ogni passo le matrici tempo-varianti del problema QP (b e F, rispettivamente 2.29 e 2.22) e risolvendo il problema QP tramite l'ottimizzatore qpOASES ([19]), trovando così la variazione ottima d'ingresso; infine, dopo aver applicato al sistema il primo campione della sequenza ottima trovata, procede all'aggiornamento dello stato del sistema stesso e dell'ingresso complessivo, salvando quest'ultimo e le uscite del sistema.
In questa funzione, così come nella precedente, è prevista la possibilità di integrare una funzione di matching del riferimento, che opera secondo quanto descritto in RIF!!!

Riferimenti bibliografici

[1] D'Ambrosio D., *Algoritmi di motion cueing per simulatori di veicolo*, Tesi di Laurea Magistrale in Ingegneria dell'Automazione, Università di Padova, 2010

[2] Pozzi M., *Motion Cueing per simulatori dinamici di veicolo: algoritmi MPC basati su modelli percettivi*, Tesi di Laurea Magistrale in Ingegneria dell'Automazione, Università di Padova, 2010

[3] Augusto B. D. C. , Loureiro R. J. L., *Motion Cueing in the Chalmers Driving Simulator: A Model Predictive Control Approach*, Master of Science Thesis, Chalmers University of Technology, Göteborg, Sweden, 2009

[4] F. Colombet, M. Dagdelen, G. Reymond, *Motion Cueing: What is the impact on the driver's behavior?*. Monaco, 2008.

[5] M. Dagdelen, G. Reymond, A. Kemeny *Model-based predictive motion cueing strategy for vehicle driving simulators*. Guyancourt, 2009

[6] R.V. Parrish, J.E. Dieudonne, *Coordinated adaptive washout for motion simulators*. Journal of Aircraft, 1975

[7] Gilles Reymond, Andras Kemeny, *Motion Cueing in the Renault Driving Simulator*. 2000.

[8] M. Wentink, W. Bles, R. Hosman, *Design & evaluation of spherical washout algorithm for Desdemona simulator*. Soesterberg, 2005.

[9] L.D. Reid, M.A. Nahon, *Flight simulation motion-base drive algorithms: part 1 - developing and testing the equations*. Toronto, 1985

[10] R. Sivan, J. Ish shalom, *An optimal control approach to the design of moving flight simulators*. IEEE International COnference on Systems,Man and Cybernetics, 1982

[11] Wang L., *Model predictive control system design and implementation using MATLAB*, Springer, 2009

[12] Maciejowski J. M., *Predictive control with constraints*, Prentice Hall, 2002

[13] Dagdelen M., Reymond G., Kemeny A., Bordier M., Maizi N., *Model - based predictive motion cueing strategy for vehicle driving simulators*, Control Engineering Practice 17 (2009) 995-1003

[14] Bemporad A., Morari M., Ricker N. L., *Model Predictive Control Toolbox 3 User's Guide*, The MathWorks, Inc. 2009

[15] Morari M., Ricker N. L., *Model Predictive Control Toolbox for use with MATLAB - User's guide*, Version 1, The MathWorks, Inc. 1998

[16] Telban R. J., Cardullo F. M., *Motion cueing algorithm development: Initial investigation and redesign of the algorithms*, Technical report CR-2000-209863, Nasa, Marzo 2000

[17] Telban R. J., Cardullo F. M., *Motion cueing algorithm development: Human-centered linear and nonlinear approaches*, Technical report CR-2005-213747, Nasa, Maggio 2005

[18] http://www.vi-grade.com/

[19] http://www.kuleuven.be/optec/software/qpOASES

[20] Wiki,*Simulatori*. http://it.wikipedia.org/wiki/Simulazione.

[21] Wiki,*Simulations*. http://en.wikipedia.org/wiki/Simulation.

RINGRAZIAMENTI

Tantissime persone hanno contribuito nel sostenermi in questi anni universitari, pertanto la lista dei ringraziamenti è particolarmente lunga.
Inizio con il ringraziare il professor A. Beghi, per la disponibilità avuta e per l'opportunità di tesi offertami che non potevo desiderare migliore. Si ringraziano poi l'ing. Fabio Maran, per il tempo dedicatomi e per la puntualità e precisione mostrate negli interventi richiesti e in particolare l'ing. Mattia Bruschetta senza l'apporto del quale non si sarebbe giunti a questi risultati nel lavoro tesi.
Si ringraziano poi tutti i colleghi ing. sia quelli di automatica che di informatica, con i quali ho condiviso questo percorso formativo divertendomi e al tempo stesso imparando molto. Un ringraziamento speciale va però fatto a Mauro Pozzi con il quale più degli altri ho condiviso questi due anni di magistrali e con il quale ho condiviso in modo ottimo, sia dal punto di vista umano che professionale, questi sei mesi di tesi. Un ringraziamento a parte va fatto anche a Weaks per essermi stata vicina nei due anni di magistrale e grazie alla quale la mia opera di ingegnerizzazione è stata ultimata con successo.
Si ringraziano poi gli amici di vecchia data in particolare Bart, Ceic, Ritz per essermi sempre stati vicini in questi anni e per la pazienza che hanno sempre nei miei confronti. Gli amici di media data, Zaghetto a Ale Pizz, per le serate play e scacchi che mi hanno sempre permesso di svagarmi durante questi ultimi mesi. Gli amici nuovi, Gabriel e Massi, per le partitine a tennis il sabato mattina e le battaglie a scopone in treno, anche all'alba.
Si ringraziano poi Albi, il Toscanaccio, Hellas e Drak per aver condiviso con me anni di appartamento a pd, per avermi sopportato e averli resi un'esperienza unica; indimenticabili saranno le pokernight fatte in via Lovarini. In particolare si ringrazia Riccardo un amico importantissimo che è stato fondamentale per la mia crescita formativa e con il quale ho condiviso anche anni di felici attività sportive.
I miei ringraziamenti più profondi vanno fatti poi nei confronti di Federica Pizz che ha reso gli ultimi quattro mesi della mia vita, potenzialmente stressanti e duri, tra i più felici e tranquilli degli ultimi anni ed è grazie a lei che la produttività in questi ultimi mesi è stata ottima.
Scontati poi i ringraziamenti a tutta la mia famiglia, in particolare a mamma Cicc e alla sorellina Egin le quali mi hanno supportato in tutti i modi, chi da un metro chi da chilometri di distanza, senza le quali non sarei giunto a questi risultati.
Un grazie sincero a tutti voi insomma e più in generale a tutti quelli che mi sono stati vicini e hanno creduto in me!!