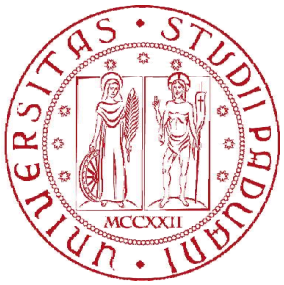

UNIVERSITÀ DEGLI STUDI DI PADOVA

Dipartimento di Ingegneria dell' Informazione

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA DELL'AUTOMAZIONE

Applicazione di *Model Predictive Control* al problema di *Motion Cueing* vincolato di un simulatore di guida

Candidato:

Luca Tosetto

Relatore:

prof. Francesco Ticozzi

Correlatori:

dott. ing. Mattia Bruschetta, dott. ing. Fabio Maran

Anno Accademico 2013-2014

Si ringrazia per la fondamentale collaborazione il dott. ing. Mattia Bruschetta e il dott. ing. Fabio Maran, del Dipartimento di Ingegneria dell'Informazione dell'Università degli Studi di Padova

Stat Roma pristina nomine, nomina nuda tenemus
"La Roma, che era, ora esiste solo nel nome, noi possediamo soltanto nudi nomi"

Bernardo di Cluny, *De contemptu mundi*, "Il disprezzo del mondo"

Indice

Parte I
Introduzione al Model Predictive Control (MPC)

1 Introduzione

Il *Model Predictive Control* (MPC) non indica una specifica strategia di controllo ma piuttosto un' ampia gamma di metodi di controllo sviluppati a partire dalla fine degli anni '70. Questi metodi fanno uso di un modello del processo per ottenere il segnale di controllo minimizzando una funzione obiettivo. Le idee alla base di MPC sono:

1. utilizzo di un modello per predire l'uscita del processo in intervalli futuri.
2. Calcolo di una sequenza di controllo per minimizzare una funzione obiettivo su un orizzonte temporale finito N. Tale calcolo viene svolto "in linea" e dunque la capacità di calcolo è una risorsa critica.
3. *Receding strategy*, cioè ad ogni dato istante t si calcola l'intera sequenza di controllo su $[t, t+N]$, ma viene utilizzato solo il primo elemento della sequenza di controllo calcolata.
4. Viene effettuata una sorta di retroazione delle uscite reali del sistema, ottenendo un errore di predizione da minimizzare.

I vari algoritmi MPC differiscono per il modello usato per rappresentare il processo e per la funzione di costo da minimizzare.

I principali vantaggi di MPC sono:

- consente di includere dei vincoli nel problema di ottimizzazione.
- E' applicabile a sistemi multivariabili con formulazione analoga a quella usata per i sistemi SISO.
- Può essere usato, scegliendo opportunamente il modello e i vincoli, per risolvere una grande varietà di problemi.

Tuttavia questo metodo presenta alcuni svantaggi:

- è necessario un modello che descriva adeguatamente il processo ma che al tempo stesso sia sufficientemente semplice da consentire i calcoli richiesti in tempo reale.
- La derivazione della legge di controllo è più complessa di quella di controllori classici (ad esempio PID).
- Nel caso di controllo adattativo, o in presenza di vincoli, il costo computazionale può essere elevato. Ciò è un problema perché i calcoli vengono svolti in tempo reale.

2 Strategia di funzionamento di MPC

In questo lavoro considereremo un sistema a tempo discreto (ma tutto vale in maniera analoga anche a tempo continuo):

$$\begin{aligned} x(t+1) &= f(x(t), u(t)), \\ y(t) &= g(x(t)), \end{aligned}$$

in cui $x(t) \in R^n$, $u(t) \in R^m$, e si prenda la seguente funzione costo generale:

$$J(y(t), x(t), u(t)) \geq 0.$$

Siamo ora in grado di descrivere formalmente la procedura MPC:

1. ad ogni istante t viene considerato un orizzonte temporale di dimensione N, definito *orizzonte di predizione*, che va da t a $t+N$, e le uscite nell'orizzonte sono predette usando il modello del processo. Queste uscite predette $\hat{y}(t+k|t)$, per $k = 1, .., N$ dipendono dai valori dello stato e dell'uscita noti all'istante t (infatti se gli stati vengono stimati occorre disporre delle uscite del sistema) e dai segnali di controllo futuri $u(t+k|t)$, per $k = 0, .., N-1$ (questi valori sono incogniti).

2. I segnali di controllo vengono calcolati ottimizzando una determinata funzione costo, tipicamente quadratica, allo scopo di mantenere il processo più vicino possibile a una data traiettoria $w(t)$ (che costituisce il riferimento da inseguire). Nella maggior parte dei casi, la traiettoria $w(t)$ è l'uscita di un pre-filtro con ingresso un segnale $r(t)$, allo scopo di rendere il segnale $w(t)$ più smussato e rallentarne l'evoluzione.

3. Viene utilizzato solo il primo elemento della sequenza di segnali di controllo, cioè $u(t|t)$, mentre gli altri segnali di controllo sono scartati. Si ottiene così lo stato e l'uscita del sistema al tempo $t+1$ e si sposta in avanti di una unità temporale l'orizzonte di predizione (da $(t, t+N)$ si passa a $(t+1, t+1+N)$). A questo punto si ripete la procedura per gli istanti successivi.

Quanto spiegato può essere descritto dal grafico in figura:

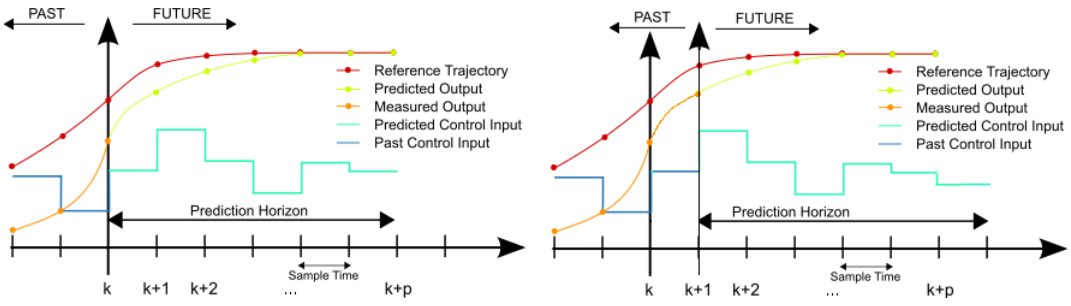

Figura 1: *schema di funzionamento di MPC*

Si nota come l'orizzonte di predizione venga spostato in avanti ad ogni iterazione, ottenendo un nuovo valore dell'uscita misurata, e anche l'uscita predetta viene ricalcolata, risultando generalmente diversa ad ogni iterazione.

La strategia MPC è analoga a livello logico a quella utilizzata per guidare un'automobile. Il guidatore conosce la "traiettoria di riferimento" (ovvero la strada da percorrere) da seguire in un orizzonte di controllo finito, e considerando le caratteristiche dell'auto decide quali azioni di controllo eseguire (accelerare, frenare, sterzare) per ottenere la traiettoria desiderata. Ad ogni istante viene eseguita una sola azione di controllo, e la procedura è ripetuta per la scelta della successiva azione di controllo, considerando un orizzonte che ad ogni istante viene spostato in avanti, e tenendo conto di eventuali cambiamenti nelle condizioni di guida.

Per implementare questa strategia, si utilizza la struttura mostrata nello schema seguente:

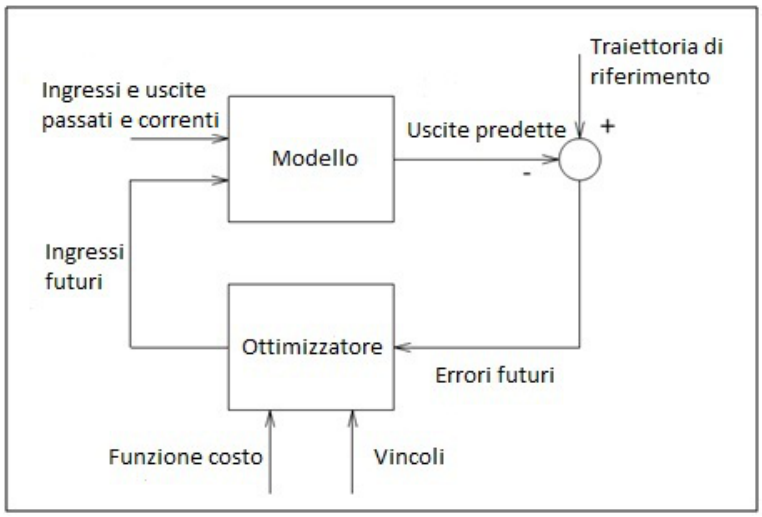

Figura 2: *schema a blocchi che illustra il flusso di dati e di segnali per MPC*

Si noti come il controllore crei un sistema in catena chiusa con retroazione dell'errore di predizione futuro.

2.1 Modello di predizione

La scelta del modello riveste un ruolo cruciale nella strategia MPC. Tale modello deve essere scelto in maniera da fornire un'adeguata descrizione della dinamica, e al tempo stesso permettere un'efficiente simulazione numerica.
Di seguito utilizzeremo un modello in forma di stato lineare a tempo discreto (a tempo continuo vale tutto in maniera analoga), del tipo:

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t). \end{aligned}$$

Con questo modello, l'uscita predetta risulta essere:

$$y(t+k|t) = Cx(t+k|t) = C[A^k x(t) + \sum_{i=1}^{k} A^{i-1} Bu(t+k-i|t)].$$

Questa descrizione può essere usata in maniera diretta anche per i processi multivariabili.

2.2 Funzione costo

Lo scopo della funzione costo è quello di assicurare l'inseguimento dell'uscita futura y nell'orizzonte considerato rispetto ad un determinato segnale di riferimento w e assicurare al tempo stesso che il controllo usato $\Delta u(t) = u(t) - u(t-1)$ sia minimo (d'ora in avanti, per non appesantire la notazione, verrà usato $\Delta u(t)$, $u(t)$ e $u(t-1)$ in luogo di $\Delta u(t|t)$, $u(t|t)$ e $u(t-1|t)$). Per ottenere queste richieste si minimizza rispetto a Δu una opportuna funzione costo, la cui espressione tipica è nella forma:

$$J(N_1, N_2, N_u) = \sum_{j=N_1}^{N_2} \delta(j)[y(t+j|t) - w(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t+j-1)]^2,$$

in cui N_1 e N_2 sono rispettivamente il minimo e il massimo orizzonte di predizione, allo scopo di definire i limiti temporali in cui è desiderabile che l'uscita segui il riferimento (in precedenza abbiamo implicitamente assunto che $N_1 = 1$, e verrà mantenuta la stessa assunzione anche nel seguito). N_u è l'orizzonte di controllo, indicante la lunghezza della sequenza di controllo da minimizzare, che può anche non coincidere con l'orizzonte di predizione (anche se solitamente i due orizzonti coincidono) e deve ovviamente rispettare la condizione $N_u < N_2 - N_1$. I coefficienti $\delta(j)$ e $\lambda(j)$ sono sequenze che considerano il comportamento futuro del sistema, e spesso assumono valori costanti. Il loro ruolo è sostanzialmente quello di "pesare", spesso in modo diverso, gli errori di predizione e gli ingressi di controllo utilizzati nella funzione costo.
Nel seguito si mostrerà che questo criterio quadratico ammette soluzione analitica se il modello è lineare e non ci sono vincoli, altrimenti deve essere usato un metodo di ottimizzazione iterativo.
L'utilizzo di $\Delta u(t)$ anziché di $u(t)$ è equivalente ad introdurre un integratore nel modello, consentendo così di annullare l'errore a regime eventualmente presente.

2.3 Formulazione del problema in forma di stato

Consideriamo il modello SISO (le formule seguenti sono estensibili al caso multivariabile in maniera del tutto analoga a quanto verrà trattato di seguito):

$$\begin{aligned} x(t+1) &= Ax(t) + Bu(t), \\ y(t) &= Cx(t). \end{aligned}$$

E' possibile definire un modello in forma di stato incrementale utilizzando l'incremento del controllo $\Delta u(t) = u(t) - u(t-1)$ al posto di $u(t)$, ottenendo così:

$$\begin{bmatrix} x(t+1) \\ u(t) \end{bmatrix} = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \begin{bmatrix} x(t) \\ u(t-1) \end{bmatrix} + \begin{bmatrix} B \\ I \end{bmatrix} \Delta u(t),$$
$$y(t) = \begin{bmatrix} C & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ u(t-1) \end{bmatrix}.$$

Definendo pertanto $\bar{x}(t) = [x(t)\, u(t-1)]^T$, il modello assume la forma:

$$\bar{x}(t+1) = M\bar{x}(t) + N\Delta u(t),$$
$$y(t) = Q\bar{x}(t).$$

Utilizzando questo modello, è possibile calcolare le uscite predette come:

$$y(t+j) = QM^j\bar{x}(t) + \sum_{i=0}^{j-1} QM^{j-i-1}N\Delta u(t+j).$$

Se il vettore di stato non è accessibile, allora occorre inserire un osservatore dello stato.
Le predizioni sull'orizzonte (supponendo per semplicità computazionale che $N_1 = 1$) sono date da:

$$y = \begin{bmatrix} \hat{y}(t+1|t) \\ \hat{y}(t+2|t) \\ \dots \\ \hat{y}(t+N_2|t) \end{bmatrix} =$$
$$\begin{bmatrix} QM\bar{x}(t) + QN\Delta u(t) \\ QM^2\bar{x}(t) + \sum_{i=0}^{1} QM^{1-i}N\Delta u(t+i) \\ \dots \\ QM^{N_2}\bar{x}(t) + \sum_{i=0}^{N_2-1} QM^{N_2-1-i}N\Delta u(t+i) \end{bmatrix},$$

che può essere espresso in forma vettoriale come:

$$y = F\bar{x}(t) + Hu,$$

con $u = [\Delta u(t)\, \Delta u(t+1) \dots \Delta u(t+N_u-1)]^T$, H è una matrice triangolare inferiore a blocchi i cui elementi non nulli sono dati da $H_{ij} = QM^{i-j}N$, mentre $F = \begin{bmatrix} QM \\ QM^2 \\ \dots \\ QM^{N_2} \end{bmatrix}$.
Si nota come il modello il forma vettoriale sia composto da una prima parte dipendente dallo stato corrente e noto all'istante t, mentre la seconda parte dipenda dal vettore dei controlli futuri u, che costituisce la variabile decisionale da calcolare.

La sequenza di controllo u è calcolata minimizzando la funzione obiettivo, la quale (nel caso in cui $\delta(j) = 1$ e $\lambda(j) = \lambda$)può essere riscritta come:

$$J = (Hu + F\bar{x}(t) - w)^T(Hu + F\bar{x}(t) - w) + \lambda u^T u,$$

in cui w è un vettore contenente i valori del segnale di riferimento $w(t)$ nell'orizzonte di predizione. Se non sono presenti vincoli, esiste una soluzione analitica data da:

$$u = (H^T H + \lambda I)^{-1} H^T (w - F\bar{x}(t)).$$

Solo il primo elemento della sequenza di controllo di u, cioè $\Delta u(t)$, è utilizzato nel controllo, e l'intero calcolo è ripetuto nel successivo istante di campionamento, con lo stato $x(t+1)$ stimato se necessario dall'uscita misurata del sistema.

Il tutto può essere espresso dal seguente schema a blocchi:

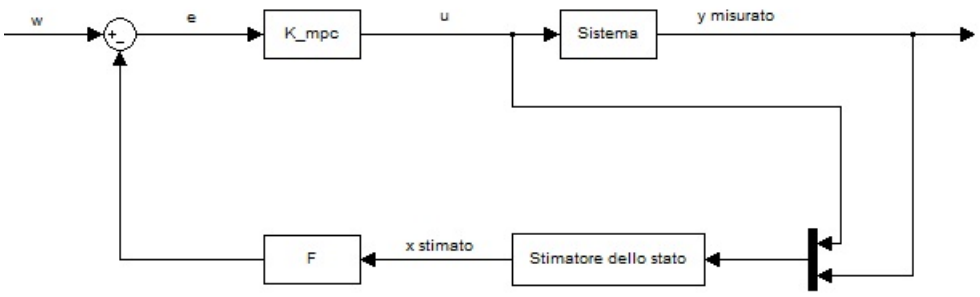

Figura 3: *schema a blocchi per implementare MPC in assenza di vincoli*

in cui `e` indica la quantità $w - F\bar{x}(t)$, il blocco `K_mpc` indica la matrice $(H^T H + \lambda I)^{-1} H^T$, `u` indica il primo elemento della sequenza di controllo u, mentre il blocco `F` rappresenta la matrice F. In questo schema è stato implementato uno stimatore di ordine intero e in catena chiusa, che necessita sia delle uscite sia dell'ingresso.

Effettuando la sostituzione $G = 2(H^T H + \lambda I)$, $b = (F\bar{x}(t) - w)^T H$, $f_0 = (F\bar{x}(t) - w)^T (F\bar{x}(t) - w)$, la funzione costo diventa:

$$J = \frac{1}{2} u^T G u + b^T u + f_0,$$

la quale è chiaramente una funzione quadratica rispetto all'incognita u.
E' possibile dimostrare che quando l'orizzonte massimo di predizione e l'orizzonte di controllo tendono all'infinito, e non ci sono vincoli, il controllore predittivo diventa il regolatore quadratico lineare (LQR), in cui la sequenza di controllo ottimo è calcolata da una retroazione statica utilizzando un guadagno in retroazione ottenibile risolvendo una opportuna equazione algebrica di Riccati. Questa equivalenza si rivela fondamentale per l'analisi teorica delle proprietà di MPC, come ad esempio la stabilità in catena chiusa.

3 MPC vincolato

Finora tutti i segnali considerati non hanno limiti di ampiezza, tuttavia tutti i processi reali sono soggetti a vincoli, dovuti a ragioni di costruzione o di sicurezza. Inoltre i punti operativi dell'impianto devono soddisfare obiettivi economici esprimibili mediante vincoli. Il sistema di controllo spesso opera vicino a questi limiti, rischiando in alcuni casi una loro violazione. I vincoli si suddividono in vincoli sulle variabili di uscita e vincoli sulle variabili d'ingresso. I primi sono dovuti a motivi di sicurezza e devono essere controllati in anticipo, mentre i secondi possono essere rispettati in maniera più semplice, ad esempio utilizzando saturatori.
In generale, i vincoli possono essere vincoli di ampiezza sul segnale di controllo, limiti sulle variazioni del segnale di controllo, e limiti sull'uscita:

$$U_{min} \leq u(t) \leq U_{max} \, \forall t,$$
$$u_{min} \leq u(t) - u(t-1) \leq u_{max} \, \forall t,$$
$$y_{min} \leq y(t) \leq y_{max} \, \forall t.$$

Di seguito vengono elencati alcuni tipi di vincolo:

- **vincoli di banda:** in alcuni casi è richiesto che le variabili controllare seguano una traiettoria entro una certa fascia (ad esempio un profilo di temperatura da seguire con una certa tolleranza). In tal caso il vincolo risulta $y_{min}(t) \leq y(t) \leq y_{max}(t)$.
- **Vincoli di sovra elongazione:** in alcuni processi l'overshoot non è desiderabile, ad esempio nel caso di manipolatori. In tal caso ogni volta si produce un cambiamento nell'uscita, ritenuta costante nel periodo considerato, si aggiungono vincoli del tipo $y(t+k) \leq w(t)$, in cui k indica l'istante in cui può verificarsi l'overshoot.
- **Non linearità dell'attuatore:** molti attuatori hanno dead-zones e altri tipi di non linearità. Questi vincoli non lineari possono essere tipicamente introdotti nel modello in forma di disuguaglianze, per esempio:

$$\begin{aligned} U_{min} &\leq u(t)\, \forall t, \\ u(t) &\leq U_{max}\, \forall t, \\ u_{min} &\leq u(t) - u(t-1)\, \forall t, \\ u(t) - u(t-1) &\leq u_{max}\, \forall t. \end{aligned}$$

  La regione ammissibile generata da questi vincoli in generale non è convessa, e pertanto in questo caso la risoluzione del problema risulta più difficile (non possono essere utilizzati gli strumenti validi per la risoluzione dei problemi di programmazione convessa).
- **Vincoli di insieme terminale:** in alcuni casi lo stato finale (cioè lo stato alla fine dell'orizzonte di predizione) del MPC è forzato ad appartenere ad un insieme terminale, imponendo l'uso di alcuni vincoli sul vettore di stato.

E' possibile vedere come tutti questi vincoli (ad eccezione dei vincoli di non linearità dell'attuatore) possano essere espressi in forma lineare. Ad esempio, nel caso dei vincoli sull'uscita, si può ottenere, utilizzando la notazione precedentemente introdotta, la seguente formula: $y_{min} \leq F\bar{x}(t) + Hu \leq y_{max}$.
La prima parte del membro centrale di questa disuguaglianza è dipendente direttamente dal vettore di stato, che cambia in genere ad ogni intervallo di campionamento e che pertanto deve essere ricalcolata opportunamente ad ogni iterazione.

3.1 Vincoli e ottimizzazione

Da quanto visto finora, il MPC vincolato può essere spesso ricondotto a un problema di programmazione quadratica con vincoli lineari (il quale è un caso particolare del problema di programmazione convessa). In questa sezione verranno descritti alcuni metodi iterativi per la risoluzione di questa tipologia di problema.
Non ci si addentrerà nei dettagli tecnici, per tali dettagli si rimanda il lettore al testo di Liuping Wang: *Model Predictive Control System Design and Implementation Using MATLAB*, cap. 2, e al testo di J.M.Maciejowski: *Predictive Control with Constraints*, cap. 3.

3.1.1 Metodo *Active Set*

Consideriamo un problema generale di programmazione quadratica:

$$\min \frac{1}{2} u^T G u + b^T u + f_0, \tag{1}$$

con vincoli: $Au = a$, $Bu \leq \omega$. Si assume che sia disponibile una soluzione iniziale u^0 ammissibile (si ritornerà in seguito su come ottenere questa soluzione).
Per questa soluzione, tutti i vincoli sono soddisfatti.
I vincoli di disuguaglianza comprendono alcuni vincoli validi con la disuguaglianza, e

altri validi con l'uguaglianza, cioè vincoli attivi. Questi ultimi definiscono il cosiddetto *active set*.
Questo *active set* può essere perciò definito come $B_u = \omega_u$, in cui il pedice indica le righe di $Bu \leq \omega$ corrispondenti ai vincoli attivi.

Data una un'iniziale soluzione ammissibile $\mathbf{u}^0$ (in seguito verrà spiegato come ottenere questa soluzione), si risolve il problema di minimizzazione (1) nell'incognita $d^1 = u^0 + s$, sottoposto ai vincoli dell'*active set* $B_u d^1 = \omega_u$ e ai vincoli di uguaglianza $Ad^1 = a$.

A questo punto, se d^1 non è ammissibile per l'insieme dei vincoli del problema di partenza dato da $Ad^1 = a$ e $Bd^1 \leq \omega$, allora qualche vincolo inattivo diventa attivo nel punto $u^0 + \alpha s$ per qualche costante α. Una volta trovata questa costante α (ad esempio con un algoritmo *line search*), il punto iniziale per la nuova iterazione diventa $u^1 = u^0 + \alpha s$, e all'insieme dei vincoli attivi si aggiunge il vincolo attivato da u^1.

Se invece d^1 è ammissibile, allora occorre effettuare un test di ottimalità per controllare se è un minimo globale. Se il test dà risultato positivo, allora l'algoritmo si arresta, mentre se la soluzione $d^1 = u^0 + s$ non è un minimo globale allora è possibile trovare una soluzione migliore eliminando dall' *active set* alcuni vincoli (per la scelta di quali vincoli eliminare, come pure per la descrizione di alcuni test di ottimalità, si rimanda al testo di Maciejowski).
L'algoritmo a questo punto può procedere iterativamente con il nuovo punto iniziale e i nuovi vincoli.

Per ottenere una soluzione iniziale ammissibile si può procedere come segue: si suppone di avere una soluzione non ammissibile u^0 che soddisfa i vincoli di uguaglianza $Au^0 = a$ e soddisfa solo un sottoinsieme dei vincoli di disuguaglianza: $B_s u^0 = \omega_s$, mentre gli altri vincoli non sono soddisfatti: $B_u u^0 > \omega_u$. Allora si cerca una soluzione u che risolve il seguente problema (B_u^j indica le righe della matrice B_u alle quali corrispondono vincoli di disuguaglianza non soddisfatti, e analogamente ω_u^j indica gli elementi di ω_u corrispondenti ai vincoli di disuguaglianza non soddisfatti):

$$\min \sum_j (\omega_u^j - B_u^j u), \tag{2}$$

con vincoli: $Au = a$, $B_s u = \omega_s$. Questo problema risulta essere un problema di programmazione lineare con vincoli lineari, per il quale può essere usato l'algoritmo del simplesso. Questo metodo presenta tuttavia alcuni inconvenienti:

- La soluzione ammissibile che si trova è arbitraria, e può essere molto lontana dalla soluzione ottimale del problema quadratico dal quale siamo partiti.
- L'algoritmo del simplesso richiede un numero di vincoli attivi pari alla dimensione di u, perciò può non essere possibile utilizzare la soluzione ottenuta dal problema di programmazione quadratica come soluzione non ammissibile da cui partire.

Questi problemi possono essere superati aggiungendo un multiplo della funzione costo del problema quadratico (1) alla funzione (2). Questo "sposta" la soluzione ammissibile di partenza verso la soluzione ottima del problema quadratico, e trasforma un problema lineare in un problema quadratico con vincoli di uguaglianza, che è risolubile in maniera indipendente dal numero dei vincoli.

Per la risoluzione di un problema quadratico con vincoli di uguaglianza si può utilizzare il seguente metodo: sia $Au = a$ l'insieme dei vincoli, con $A \in R^{m*n}$, $u \in R^n$, $a \in R^m$, $n > m$, si esprimono m elementi di u in funzione dei rimanenti $n - m$ elementi e li si

sostituiscono alla funzione da minimizzare, ottenendo così un problema di ottimizzazione in $n - m$ incognite e senza vincoli.

Ad ogni iterazione si sceglie un punto che migliora la soluzione all'iterazione precedente, e pertanto, a causa della convessità del problema, alla fine questo metodo assicura l'ottenimento di un minimo globale.

3.1.2 Metodo a punto interno (metodo della barriera logaritmica)

L'algoritmo *active set* nel caso peggiore può avere complessità esponenziale nel numero dei parametri, cioè il numero di vincoli e variabili, mentre il metodo a punto interno ha complessità polinomiale. Il metodo a punto interno che verrà trattato in seguito è il cosiddetto "metodo della barriera logaritmica". Il metodo *active set* ricerca i punti sulla frontiera della regione ammissibile, mentre in questo metodo ad ogni iterazione si ottengono sempre soluzioni ammissibili del problema di ottimizzazione, e si cerca di penalizzare le soluzioni che si avvicinano alla frontiera della regione ammissibile. Data una funzione quadratica $f(u)$ da minimizzare rispetto ad u, soggetta ai vincoli $Au = a$, $B_i u < \omega_i$, si risolve il problema ausiliario:

$$\min f(u) + c \sum_i -log(-B_i u + \omega_i),$$

sottoposto ai vincoli $Au = a$.
Ancora una volta, in analogia a quanto scritto sopra, è possibile esprimere m elementi di u in funzione dei rimanenti $n - m$ e sostituirli nella funzione, come spiegato sopra, ottenendo un problema senza vincoli.
Se la funzione obiettivo $f(u) + c \sum_i -log(-B_i u + \omega_i)$ è convessa e differenziabile, allora è possibile utilizzare l'algoritmo di Newton per la risoluzione.

Questo problema è una approssimazione del problema di partenza, e la bontà dell'approssimazione dipende dalla scelta della costante c. Per assicurare una buona approssimazione, la variabile c viene incrementata secondo la regola $c^{k+1} = c^k \mu$, con $\mu < 1$, ad ogni iterazione k.
E' possibile dimostrare che se $f(u)$, $B_i u - \omega_i$, $\sum_i -log(-B_i u + \omega_i)$ sono funzioni convesse sulla regione ammissibile, anch'essa convessa, allora la successione dei punti minimi del problema ausiliario converge a un minimo globale.

4 Stabilità

Il controllo predittivo è una tecnica di controllo in retroazione, e pertanto presenta indiscutibili vantaggi rispetto ad altre strategie di controllo, tuttavia permane il rischio che il sistema in catena chiusa sia instabile. Anche se le prestazioni dell'impianto vengono ottimizzate sull'orizzonte di predizione, e anche se l'ottimizzazione viene iterata, ogni ottimizzazione non si cura di cosa succeda oltre l'orizzonte di predizione, e perciò l'impianto può venir portato in uno stato instabile. L'analisi della stabilità diventa fondamentale quando il controllore viene ricalcolato ad ogni iterazione, come ad esempio nel controllo adattativo.

Osservando attentamente la formula per il calcolo della sequenza di ingresso ottima ottenuta alla fine della Sezione 2.3, è possibile osservare che in assenza di vincoli e con riferimento nullo il problema MPC viene risolto da un controllore lineare che consiste in una retroazione dallo stato, e perciò l'analisi della stabilità in catena chiusa è particolarmente agevole, basta infatti analizzare i poli del sistema in catena chiusa. Se invece, oltre all'assenza di

vincoli, si suppone che l'orizzonte di predizione e di controllo siano infiniti, allora il problema si riduce ad un problema LQR, e l'analisi di stabilità anche in questo caso diventa agevole grazie a risultati già noti per i problemi di controllo ottimo (a tal proposito si veda il testo di E.Fornasini: *Appunti di Teoria dei Sistemi*, cap.10).

4.1 Stabilità con orizzonte infinito in assenza di vincoli

Consideriamo l'assenza di vincoli nel problema di ottimizzazione di J, e che il modello del processo sia perfetto. Se l'orizzonte (di predizione e controllo) è infinito, vale il principio di ottimalità di Bellman, cioè la traiettoria ottima calcolata ad una certa iterazione k è identica alla sezione di traiettoria calcolata all'iterazione $k+1$ e troncata al valore k-esimo. Questo principio non si applica al caso con orizzonte finito, perché ad ogni iterazione si ottiene un problema di ottimizzazione diverso, e la traiettoria ottima può essere completamente differente da quella calcolata nella iterazione precedente.
L'orizzonte infinito perciò assicura che la funzione costo decresca ad ogni iterazione (nell'ipotesi che non ci siano disturbi e che il modello sia perfetto) e pertanto la funzione costo può essere usata come funzione di Lyapunov per l'analisi della stabilità. Perciò il teorema di stabilità di Lyapunov assicura che l'origine è asintoticamente stabile. Per maggiori dettagli circa la dimostrazione di questo risultato si rimanda al testo di J.M.Maciejowski: *Predictive Control with Constraints*, cap. 6.

4.2 Stabilità con orizzonte finito in presenza di vincoli

Nel testo di James B. Rowlings e David Q. Mayne: *Model Predictive Control: Theory and Design* vengono presentate alcune assunzioni e ipotesi che assicurano la stabilità in catena chiusa in presenza di vincoli e con orizzonte finito. Occorre tuttavia modificare leggermente la notazione. Il riferimento è nullo (in questo caso la stabilità è intesa come stabilità nell'origine), mentre la funzione costo viene ora descritta dalla seguente formula:

$$J(t) = \sum_{j=0}^{N-1} l(x(t+j), u(t+j-1) + J_f(x(t+N))),$$

in cui $l(x(\cdot),u(\cdot))$ è una generica funzione costo che dipende dallo stato e dall'ingresso, mentre $J_f(x(N))$ dipende soltanto dallo stato alla fine dell'orizzonte di predizione (che va da $t = 0$ a $t = N$).
Si definiscano inoltre X come la regione ammissibile per lo stato (ottenuta dai vincoli sull'uscita, che possono essere tradotti in vincoli sullo stato), X_f l'insieme a cui appartiene lo stato finale $x(N)$, U la regione ammissibile per l'ingresso, X_N il sottoinsieme di X per il quale esiste una soluzione al problema di minimizzazione vincolata di J.

Ora occorre fare alcune assunzioni:

1. il modello deve essere controllabile e osservabile.
2. L'equazione che descrive il modello e la funzione costo sono continue, e in zero valgono entrambe zero.
3. X, $X_f \subseteq X$, U sono insiemi chiusi, compatti e contenenti l'origine.
4. Lo stato terminale è vincolato a stare in X_f.
5. X_f è un invariante di controllo per il sistema che descrive il modello.
6. $l(x(\cdot), u(\cdot)) \geq \alpha_1(|x(\cdot)|) \forall x(\cdot) \in X_N, \forall u(\cdot) \in U$, in cui $\alpha_1(\cdot)$ è una funzione K_∞ (continua, strettamente crescente, nulla in zero, non limitata).
7. $J_f \leq \alpha_2(|x(\cdot)|) \forall x(\cdot) \in X_f$, in cui $\alpha_2(\cdot)$ è anch'essa una funzione K_∞.

Allora, sotto tutte queste ipotesi, l'origine è asintoticamente stabile con regione di attrazione data da X_N.

La dimostrazione di questo teorema si può trovare nel suddetto testo di James B. Rowlings e David Q. Mayne: *Model Predictive Control: Theory and Design* cap. 2.

5 *Tuning*

Esistono molti parametri modificabili nel MPC:

- pesi,
- modellizzazione del disturbo e dell'osservatore,
- traiettoria di riferimento.

Quando si effettua *tuning* su MPC si dovrebbero considerare le prestazioni in presenza di vincoli. Tuttavia ci sono pochi risultati teorici applicabili in questo contesto, pertanto in tali casi ci si affida spesso a simulazioni con vincoli attivi. In questa sezione i vincoli non verranno considerati, e questo ci porta ad analizzare un controllore lineare consentendo di fornire alcuni risultati teorici.

5.1 Effetti dei pesi di controllo

Aumentare i pesi $\lambda(j)$ del controllo rispetto ai pesi $\delta(j)$ sull'errore di inseguimento ha come effetto la riduzione dell'attività di controllo. Un aumento indefinito di questi pesi porta l'attività di controllo a zero, il che corrisponde a disattivare l'azione di retroazione. Pertanto se l'impianto è stabile allora un incremento adeguato dei pesi di controllo mantiene la catena chiusa stabile. D'altra parte ciò provoca una lenta risposta ai disturbi, perché le azioni di controllo sono penalizzate. Se l'impianto non è stabile allora tendenzialmente la retroazione resterà instabile.

5.2 Dinamica dello stimatore

Si consideri nuovamente il seguente schema a blocchi:

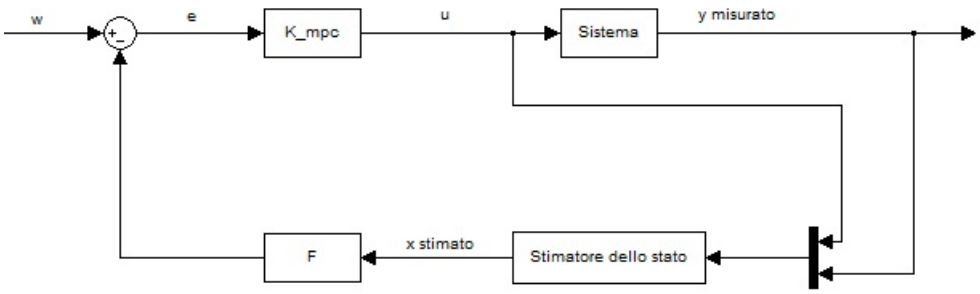

Figura 4: *schema a blocchi che descrive il funzionamento di MPC in assenza di vincoli*

Tipicamente lo stimatore utilizzato è a catena chiusa e di ordine intero cioè utilizza l' uscita misurata e l'ingresso utilizzato per produrre la stima, e ha la stessa dimensione dello stato. Perciò, utilizzando il sistema descritto nella Sezione 2:

$$\bar{x}(t+1) = M\bar{x}(t) + N\Delta u(t),$$
$$y(t) = Q\bar{x}(t),$$

e introducendo una matrice L di dimensioni pari a quelle di Q^T, lo stato stimato $\hat{\hat{x}}(t)$ è ottenuto dalla seguente equazione:

$$\hat{\hat{x}}(t+1) = (M + LQ)\hat{\hat{x}}(t) - Ly_m(t) + N\Delta u(t),$$

in cui y_m indica l'uscita misurata.
Pertanto lo schema a blocchi risulta essere:

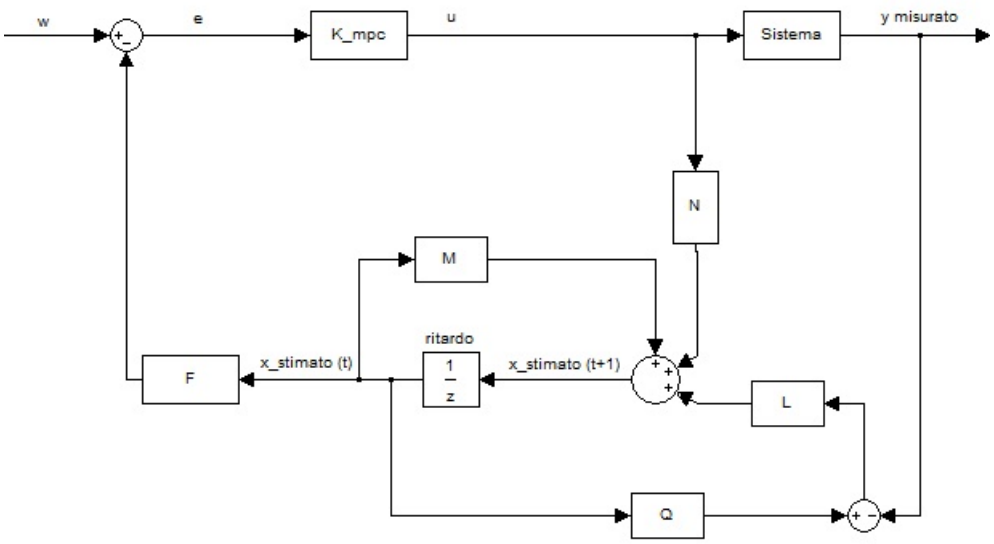

Figura 5: *schema a blocchi con inserimento di uno stimatore dello stato*

L'uscita misurata è legata allo stato $\hat{\hat{x}}(t)$ (indicato nello schema come `x_stimato (t)`) dalla seguente funzione di trasferimento:

$$\hat{\hat{X}}(z) = -[zI - (M + LQ)]^{-1}LY_m(z).$$

Ciò implica che l'uscita misurata è filtrata prima di agire sul controllore, e i poli di questo filtro sono gli autovalori dello stimatore. Tali autovalori pertanto sono determinanti nella risposta ai disturbi che agiscono sull'uscita e nella stabilità in catena chiusa.

5.3 Modellizzazione del disturbo

Se un disturbo deterministico e persistente agisce sul sistema (ad esempio un segnale costante, una rampa o una sinusoide), tale segnale può essere descritto mediante un opportuno modello, creando così un modello aumentato dell'intero sistema.
Ad esempio, se il disturbo appartiene alla seguente classe di disturbi:

$$\begin{aligned} x_d(t+1) &= A_d x_d(t), \\ d(t) &= C_d x_d(t). \end{aligned}$$

Mentre lo stato del sistema è legato dal disturbo dalla seguente equazione:

$$x(t+1) = Ax(t) + Dd(t) + Bu(t)$$

Allora è possibile definire un sistema con stato aumentato:

$$\begin{bmatrix} x(t+1) \\ x_d(t+1) \end{bmatrix} = \begin{bmatrix} A & DC_d \\ 0 & A_d \end{bmatrix} \begin{bmatrix} x(t) \\ x_d(t) \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u(t).$$

La funzione costo deve essere adattata di conseguenza, ad esempio trascurando le ultime componenti x_d dello stato del sistema, che corrispondono alla dinamica del disturbo.

Si supponga che lo stato sia noto, che gli orizzonti di predizione e di controllo siano infiniti, che non ci siano vincoli e che il riferimento sia a zero; come già spiegato in precedenza il controllo da utilizzare con queste ipotesi è un controllo LQR, descrivibile dal seguente schema a blocchi:

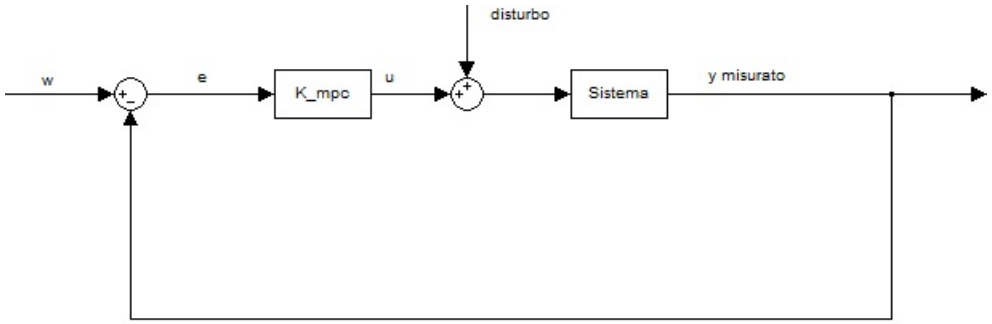

Figura 6: *schema per implementazione MPC con aggiunta di un disturbo*

inoltre sia $D(z) = \frac{N_d(z)}{D_d(z)}$ la trasformata zeta del disturbo.
Allora, per il principio del modello interno, condizione sufficiente per la reiezione del disturbo è che il denominatore del controllore `K_mpc` contenga il polinomio $D_d(z)$. Tutto questo si può ottenere aggiungendo ai poli di `K_mpc` gli zeri di $D_d(z)$. Ricordando che la funzione di trasferimento di `K_mpc` è data da $(H^T H + \lambda I)^{-1} H^T$, questo equivale ad aggiungere gli zeri di $D_d(z)$ agli zeri del polinomio caratteristico di $(H^T H + \lambda I)$.

5.4 Traiettoria di riferimento e pre-filtro

E' facile intuire che una traiettoria di riferimento più lenta dovrebbe portare a un'azione di retroazione meno aggressiva, e quindi dovrebbe migliorare la robustezza agli errori di modellizzazione. Tuttavia in questo settore la ricerca scientifica ha analizzato solo alcuni casi particolari, e perciò non sono noti importanti risultati teorici a riguardo.
Più interessante è l'analisi del pre-filtro. Questo dispositivo è posto fuori dall'anello di retroazione, e pertanto non influenza la stabilità o la robustezza della retroazione, e non ha effetti sulla risposta ai disturbi.
Uno dei principali utilizzi di questo dispositivo è quella di trasformare la traiettoria di set-point $r(t)$ nella traiettoria di riferimento $w(t)$, al fine di ottenere una traiettoria di riferimento che consenta prestazioni migliori.
Altro compito del pre-filtro è quello di ridurre la saturazione degli attuatori, fornendo una traiettoria che limiti l'ampiezza degli errori tra la traiettoria stessa e l'uscita predetta.

Parte II
Esempio di simulatore di guida utilizzando MPC

6 Descrizione del sistema da controllare

6.1 Introduzione

Negli ultimi anni si è assistito ad un aumento di interesse nello sviluppo di simulatori di guida dinamici, utilizzati in differenti ambiti, come le corse automobilistiche (allenamento professionale alla guida, sviluppo di veicoli virtuali), sistemi per il controllo di sicurezza (onde evitare gli incidenti), riabilitazione medica.
In particolare è aumentato l'interesse per la realizzazione di piattaforme di piccole dimensioni e basso costo.
In un simulatore di guida è fondamentale riprodurre fedelmente le sensazioni del pilota, in modo che possa pienamente utilizzare l'esperienza virtuale.
A tale scopo ha un ruolo cruciale la strategia di *motion cueing* (MC), cioè l'algoritmo usato per trasformare le accelerazioni e le velocità del veicolo in comandi di movimento per la piattaforma.
Una delle principali difficoltà nel progettare algoritmi MC efficaci sta nella natura complessa dei sistemi di percezione umani, poiché da un punto di vista fisiologico il ruolo e le priorità degli stimoli nella percezione delle accelerazioni non è ben noto.
In questa tesi è descritto un algoritmo per la risoluzione di MC basato sul *Model Predictive Control* secondo la strategia descritta da Nikhil J. I. Garrett e Matthew C. Best in *Model predictive driving simulator motion cueing algorithm with actuator-based constraints*, in cui l'evoluzione delle lunghezze degli attuatori che azionano il simulatore di guida viene inserita nel modello del sistema, e ciò allo scopo di poter inserire i vincoli sul movimento della piattaforma direttamente su queste lunghezze. Allo scopo è stato opportunamente modificato un software Matlab per il *motion cueing* sviluppato dal Dipartimento di Ingegneria dell'Informazione dell'Università di Padova.

6.2 Funzionamento del *Motion Cueing*

Nella figura seguente è descritta la piattaforma utilizzata in questo studio. Si tratta di una piattaforma Stewart a 6 gradi di libertà, cioè spostamento longitudinale (lungo l'asse x), spostamento laterale (lungo l'asse y), spostamento verticale (lungo l'asse z), angolo di *pitch* (denotato con θ indica la rotazione lungo l'asse y), angolo di *roll* (denotato con ϕ indica la rotazione lungo l'asse x), angolo di *yaw* (indicato con ψ indica la rotazione lungo l'asse z):

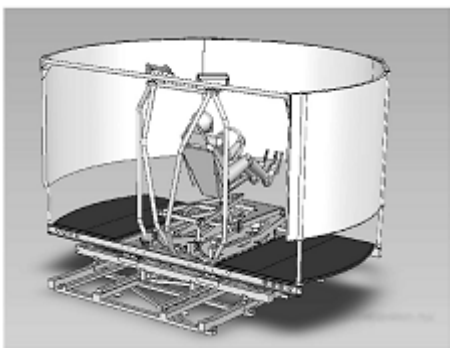

Figura 7: *Raffigurazione del simulatore utilizzato*

Il simulatore dispone di 6 attuatori posti sotto la piattaforma, che forniscono il movimento richiesto variando la loro lunghezza.
L'utilizzo di attuatori lineari consente di ottenere risultati soddisfacenti nella simulazione anche con hardware di ridotte dimensioni. Lo schermo del simulatore copre più di 180 gradi e si muove in accordo con la piattaforma per garantire una piena immersione nell'ambiente virtuale. Infine sono presenti delle forze di reazione sul volante e sul sistema frenante per rafforzare le sensazioni del pilota durante la guida.
Lo schema concettuale del funzionamento di MC è riportata nella figura seguente:

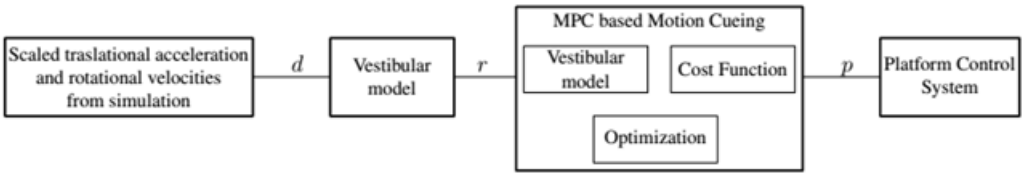

Figura 8: *schema di funzionamento del motion cueing*

Esso è composto dai seguenti passi:

- ottenere le uscite attuali d del veicolo simulato, cioè le accelerazioni lungo gli assi e le velocità angolari della piattaforma,
- ottenere le accelerazioni e le velocità angolari percepite r filtrando d col modello del sistema vestibolare (quest'ultimo verrà spiegato nel dettaglio nelle prossime Sezioni), producendo così il riferimento per l'algoritmo MPC,
- calcolare via MPC il segnale di spostamento p da inviare al sistema di controllo della piattaforma per ottenere il comportamento percepito desiderato.

La maggior parte degli algoritmi di MC utilizzati impone i vincoli direttamente sullo spazio operativo, mentre l'algoritmo qui presentato utilizza la lunghezza e la velocità degli attuatori come vincoli del problema. Tuttavia, anziché considerare un controllo che valuti assieme tutti i 6 gradi di libertà del dispositivo, si è preferito (per motivi computazionali) realizzare un controllo per la coppia di gradi di libertà spostamento longitudinale/*pitch*, un altro per la coppia spostamento laterale/*roll*, uno per lo spostamento verticale, e uno per l'angolo di *yaw*. Ciò che cambia nella realizzazione di questi controllori sono: il modello vestibolare usato e i gradi di libertà controllati.

6.3 Modello vestibolare utilizzato dai controllori

Nella prima parte della Sezione verrà brevemente esposto il modello matematico generale utilizzato per descrivere il sistema vestibolare umano, mentre nelle Sottosezioni verrà descritto come adattare tale modello ai controllori utilizzati. I modelli vestibolari dei controllori forniscono i riferimenti da inseguire per l'algoritmo MPC.

6.3.1 Modello generale del sistema vestibolare umano

L'utilizzo del movimento percepito necessita di un modello vestibolare dell'orecchio umano, composto dalle otoliti (che forniscono le accelerazioni lineari percepite) e dai canali semicircolari (che forniscono le velocità angolari percepite). Dei numerosi modelli presentati, verrà utilizzato quello descritto da R. Telban, W. Wu, F. Cardullo, L. R. Center in *Motion Cueing Algorithm Development: Initial Investigation and Redesign of the Algorithms*.
E' importante considerare che tale modello è a tempo continuo, ma per i nostri scopi verrà in seguito considerata la versione campionata.

Secondo questo modello, la velocità angolare percepita $\hat{\omega}$ è legata alla velocità angolare della testa ω dalla seguente equazione che descrive in maniera approssimata il funzionamento di ognuno dei tre canali semicircolari:

$$W_{S_i}(s) = \frac{\hat{\omega}_i(s)}{\omega_i(s)} = 5.73\frac{80s^2}{(1+80s)(1+5.73s)},$$

in cui $i = x, y, z$ indicano l'asse in cui avviene la rotazione (angoli di *roll*, *pitch*, *yaw*).
La relazione tra l'accelerazione percepita $\hat{a}$ e quella reale a è data dalla seguente equazione che descrive approssimativamente il funzionamento delle otoliti:

$$W_{O_i}(s) = \frac{\hat{a}_i(s)}{a_i(s)} = 0.4\frac{1+10s}{(1+5s)(1+0.016s)},$$

in cui $i = x, y, z$ indicano l'asse in cui è diretta l'accelerazione. Il metodo MPC richiede una formulazione in spazio di stato. I seguenti sistemi costituiscono una realizzazione delle due equazioni prima definite:

$$\Sigma_{S_i} = (A_{S_i}, B_{S_i}, C_{S_i}, D_{S_i}),$$

$$\Sigma_{O_i} = (A_{O_i}, B_{O_i}, C_{O_i}, D_{O_i}),\ i = x, y, z.$$

Quando $i = x$, pertanto, il primo sistema riceve in ingresso $\dot{\phi}$ e restituisce $\hat{\dot{\phi}}$ (velocità percepita di *roll*), mentre per $i = y, z$ il sistema, rispettivamente, riceve in ingresso $\dot{\theta}$ e $\dot{\psi}$, restituendo $\hat{\dot{\theta}}$ e $\hat{\dot{\psi}}$ (velocità percepita di *pitch* e *yaw*).

Analogamente il secondo sistema, al variare di $i = x, y, z$, riceve in ingresso a_x, a_y, a_z e restituisce $\hat{a}_x, \hat{a}_y, \hat{a}_z$ (accelerazioni percepite lungo l'asse x, y, z).

Il modello delle otoliti viene modificato per introdurre l'effetto della *tilt-coordination*. Gli otoliti infatti non sono in grado di distinguere tra accelerazione longitudinale e gravitazionale, e pertanto se il sedile presenta un angolo di *pitch* (o di *roll*) diverso da zero, il pilota potrebbe avvertire una accelerazione longitudinale (o laterale) apparente.
Quindi se θ e ϕ sono gli attuali angoli di *pitch* e *roll* del sedile del simulatore, il vettore di gravità g_{TILT} del sistema non inerziale solidale al sedile è dato dalla seguente rotazione del vettore di gravità inerziale (che interessa solo l'asse z):

$$g_{TILT} = R_x(\theta) \cdot R_y(\phi) \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = \begin{bmatrix} -g\sin\theta \\ g\cos\theta\sin\phi \\ g\cos\theta\cos\phi \end{bmatrix}.$$

L'angolo di *yaw* ψ non viene considerato perché la *tilt coordination* non influenza questo grado di libertà.
Prendendo $a = [a_x\, a_y\, a_z]^T$ come l'accelerazione che deve essere fornita al pilota, si ottiene che l'accelerazione effettiva da fornire risulta essere, utilizzando l'approssimazione ai piccoli angoli:

$$\tilde{a} = a - g_{TILT} = \begin{bmatrix} a_x + g\sin\theta \\ a_y + g\cos\theta\sin\phi \\ a_z - g\cos\theta\cos\phi \end{bmatrix} \approx \begin{bmatrix} a_x + g\theta \\ a_y - g\phi \\ a_z - g \end{bmatrix}.$$

Il modello del sistema vestibolare appena descritto è di natura generale, ma i controllori non dispongono di tutti i 6 gradi di libertà utilizzati, e pertanto occorre adattare adeguatamente il modello ad ogni controllore, ottenendo 4 diversi modelli. Le prossime Sottosezioni sono dedicate proprio alla descrizione di questi ultimi.
Si vedrà in particolare che:

- il modello del sistema vestibolare del controllore per lo spostamento longitudinale/*pitch* fornisce come riferimenti per l'algoritmo MPC i seguenti segnali: accelerazione percepita lungo l'asse x, velocità angolare di *pitch* percepita. Il sistema vestibolare produce anche il segnale relativo alla posizione angolare di *pitch*, utilizzata anch'essa come riferimento (come si vedrà nella Sezione 9),
- il modello del sistema vestibolare del controllore per lo spostamento laterale/*roll* fornisce come riferimenti i seguenti segnali: accelerazione percepita lungo l'asse y, velocità angolare di *roll* percepita. Il sistema vestibolare produce anche il segnale relativo alla posizione angolare di *roll*, utilizzata anch'essa come riferimento (come si vedrà nella Sezione 9),
- il modello del sistema vestibolare del controllore per lo spostamento verticale fornisce come riferimento l'accelerazione verticale percepita,
- il modello del sistema vestibolare del controllore per l'angolo di *yaw* fornisce come riferimento la velocità angolare percepita di *yaw*.

Occorre specificare inoltre che lo spostamento lungo gli assi è riferito al centro della piattaforma, ad esempio con $\dot{x}$ verrà indicata la velocità lungo l'asse x del centro della piattaforma.

6.3.2 Modello vestibolare per il controllore longitudinale/*pitch*

Per il controllore longitudinale/*pitch* viene impiegato sia il sistema che descrive il canale semicircolare Σ_{S_y} per l'asse y sia il sistema che descrive il funzionamento delle otoliti Σ_{O_x} per l'asse x (i gradi di libertà sono lo spostamento lungo l'asse x e il movimento di *pitch*, cioè la rotazione lungo l'asse y). Per utilizzare la velocità di *pitch* come ingresso al sistema delle otoliti si ricorre ad uno stato aumentato $x_{\bar{O}_x} = [x_{O_x}\, \theta]^T$. Il sistema corrispondente è dato da $\Sigma_{\bar{O}_x} = (A_{\bar{O}_x}, B_{\bar{O}_x}, C_{\bar{O}_x}, D_{\bar{O}_x})$, con

$$A_{\bar{O}_x} = \begin{bmatrix} A_{O_x} & \bar{B}_x \\ 0 & 0 \end{bmatrix}, B_{\bar{O}_x} = \begin{bmatrix} B_{O_x} & 0 \\ 0 & 1 \end{bmatrix}, C_{\bar{O}_x} = \begin{bmatrix} C_{O_x} & 0 \end{bmatrix}, D_{\bar{O}_x} = \begin{bmatrix} D_{O_x} & 0 \end{bmatrix},$$

in cui $\bar{B}_x = B_{O_x} \cdot g$ allo scopo di considerare l'effetto della *tilt coordination*.
Gli ingressi di questo sistema sono $u_{\bar{O}} = [a_x\, \dot{\theta}]^T$, mentre l'uscita è $y_{\bar{O}} = \hat{\dot{\theta}}$.
Per imporre correttamente i vincoli è conveniente esplicitare posizione x e velocità $\dot{\theta}$. Ciò può essere fatto modificando opportunamente le matrici, ottenendo il sistema vestibolare finale:

$$A_{vest,xy} = \begin{bmatrix} A_{S_y} & 0 & 0 \\ 0 & A_{\bar{O}_y} & 0 \\ 0 & 0 & A_I \end{bmatrix}, B_{vest,xy} = \begin{bmatrix} 0 & B_{S_y} \\ B_{\hat{O}_x} & \\ 1 & 0 \end{bmatrix},$$
$$C_{vest,xy} = \begin{bmatrix} C_{S_y} & 0 & 0 \\ 0 & C_{\bar{O}_x} & 0 \\ 0 & 0 & I_3 \\ 0 & 0 & 0 \end{bmatrix}, D_{vest,xy} = \begin{bmatrix} 0 & D_{S_y} \\ D_{\bar{O}_x} & \\ 0 & \\ 0 & 1 \end{bmatrix},$$

in cui $A_I = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$. Il sistema viene in seguito discretizzato da un interpolatore di ordine zero. Lo stato di questo sistema è dato da $x_{vest,xy} = [x_{S_y}\, x_{\bar{O}_x}\, \theta\, x\, \dot{x}]^T$, l'ingresso è dato da $u_{vest_xy} = [a_x\, \dot{\theta}]^T$, l'uscita è data da $y_{vest,xy} = [\hat{\dot{\theta}}\, \hat{a}_x\, \theta x\, \dot{x}\, \dot{\theta}]^T$.

6.3.3 Modello vestibolare per il controllore laterale/*roll*

Il controllore laterale/*roll* (i cui gradi di libertà sono lo spostamento lungo l'asse y e il movimento di *roll*, cioè la rotazione lungo l'asse x) presenta un modello vestibolare

analogo a quello appena descritto, in particolare viene impiegato sia il sistema che descrive il canale semicircolare Σ_{S_x} per l'asse x sia il sistema che descrive il funzionamento delle otoliti Σ_{O_y} per l'asse y. Le matrici $(A_{vest.yx}, B_{vest.yx}, C_{vest.yx}, D_{vest.yx})$ relative al sistema vestibolare si ottengono in maniera analoga quanto visto sopra (questa volta $\bar{B}_y = B_{O_y} \cdot (-g)$).
Lo stato di questo sistema è dato da $x_{vest,yx} = [x_{S_x}\, x_{\bar{O}_y}\, \phi\, y\, \dot{y}]^T$, l'ingresso è dato da $u_{vest_yx} = [a_y\, \dot{\phi}]^T$, l'uscita è data da $y_{vest,yx} = [\hat{\dot{\phi}}\, \hat{a}_y\, \theta\, y\, \dot{y}\, \dot{\phi}]^T$.

6.3.4 Modello vestibolare per il controllore verticale

Per il controllore relativo allo spostamento verticale (l'unico grado di libertà è dato dallo spostamento lungo l'asse z) il sistema vestibolare comprende solo la parte relativa al modello Σ_{O_z} degli otoliti lungo l'asse z. Per poter imporre i vincoli è conveniente esplicitare la posizione e la velocità lungo z, aggiungendo due stati e ottenendo quindi il seguente modello:

$$A_{vest,z} = \begin{bmatrix} A_{O_z} & 0 \\ 0 & A_I \end{bmatrix}, B_{vest,z} = \begin{bmatrix} B_{O_z} \\ 0 \\ 1 \end{bmatrix}, C_{vest,z} = \begin{bmatrix} C_{O_y} & 0 \\ 0 & I_2 \end{bmatrix},$$
$$D_{vest,z} = \begin{bmatrix} D_{O_z} \\ 0 \\ 0 \end{bmatrix},$$

in cui $A_I = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$. Il sistema viene infine discretizzato mediante un interpolatore di ordine zero. Lo stato di questo sistema è dato da $x_{vest,z} = [x_{O_z}\, z\, \dot{z}]^T$, l'ingresso è dato da $u_{vest_z} = a_z$, l'uscita è data da $y_{vest,z} = [\hat{a}_z\, z\, \dot{z}]^T$.

6.3.5 Modello vestibolare per il controllore dell'angolo di *yaw*

Per il controllore dell'angolo di *yaw* l'unico grado di libertà è appunto l'angolo ψ. Il sistema vestibolare pertanto comprende sono la parte relativa al canale semicircolare dell'asse z (Σ_{S_z}). Allo scopo di esplicitare la velocità di *yaw* (fornita come ingresso) si aggiunge un nuovo stato, ottenendo così le seguenti matrici:

$$A_{vest,yaw} = \begin{bmatrix} A_{S_z} & 0 \\ 0 & 0 \end{bmatrix}, B_{vest,yaw} = \begin{bmatrix} B_{S_z} \\ 1 \end{bmatrix}, C_{vest,yaw} = \begin{bmatrix} C_{S_y} & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$
$$D_{vest,yaw} = \begin{bmatrix} D_{S_z} \\ 0 \\ 1 \end{bmatrix}.$$

Il sistema viene poi discretizzato mediante interpolatore di ordine zero. Lo stato di questo sistema è $x_{vest,yaw} = [x_{S_z}\, \dot{\psi}]^T$, l'ingresso è dato da $u_{vest_yaw} = \dot{\psi}$, l'uscita è data da $y_{vest,yaw} = [\hat{\dot{\psi}}\, \psi\, \dot{\psi}]^T$.

6.4 Calcolo delle lunghezze degli attuatori

Allo scopo di inserire i vincoli sulla lunghezza degli attuatori, è necessario riuscire a calcolare ad ogni istante la lunghezza di questi ultimi. Nel seguito verrà descritto un metodo per consentire tale calcolo.

In effetti, le lunghezze $(l_i, i = 1, \ldots, 6)$ degli attuatori possono essere calcolate direttamente dalla posizione della piattaforma espressa in coordinate cartesiane, utilizzando l'aritmetica dei vettori. Infatti definendo come R_{AB} il vettore che parte dall'origine del

sistema di riferimento posto al suolo e arriva al centro della piattaforma, A_i il vettore che parte dall'origine del sistema di riferimento posto al suolo e arriva al giunto inferiore dell'attuatore i-esimo, B_i il vettore che parte dal centro della piattaforma e arriva al giunto superiore dell'attuatore i-esimo, e L la matrice di rotazione dal sistema di riferimento posto al suolo al sistema di riferimento sulla piattaforma, vale la seguente relazione:

$$l_i = R_{AB} + LB_i - A_i.$$

La matrice di rotazione L utilizzata è quella che si ottiene facendo ruotare il sistema di riferimento secondo la combinazione $Z - Y - X$, e utilizzando una approssimazione ai piccoli angoli essa presenta la seguente struttura:

$$L = \begin{bmatrix} 1 & -\psi & \theta \\ \psi + \phi\theta & 1 & -\phi \\ \phi\psi - \theta & \phi + \psi\theta & 1 \end{bmatrix},$$

in cui ϕ è l'angolo di *roll*, θ è l'angolo di *pitch*, ψ è l'angolo di *yaw*.
Pertanto combinando queste formule si ottengono le seguenti equazioni:

$$l_i = \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}_2 = \left(\begin{bmatrix} x_{AB} \\ y_{AB} \\ z_{AB} \end{bmatrix} + \begin{bmatrix} 1 & -\psi & \theta \\ \psi + \phi\theta & 1 & -\phi \\ \phi\psi - \theta & \phi + \psi\theta & 1 \end{bmatrix} \begin{bmatrix} x_{B,i} \\ y_{B,i} \\ z_{B,i} \end{bmatrix} - \begin{bmatrix} x_{A,i} \\ y_{A,i} \\ z_{A,i} \end{bmatrix} \right)_2 =$$

$$\begin{bmatrix} x_{AB} + x_{B,i} - \psi y_{B,i} + \theta z_{B,i} - x_{A,i} \\ y_{AB} + (\psi + \phi\theta)x_{A,i} + y_{B,i} - \phi z_{B,i} - y_{A,i} \\ z_{AB} + (\phi\psi - \theta)x_{B,i} + (\phi + \psi\theta)y_{B,i} + z_{B,i} - z_{A,i} \end{bmatrix}_2.$$

Il pedice indica che viene calcolata la norma 2 del vettore.
Per calcolare la velocità occorre derivare la lunghezza rispetto al tempo:

$$\frac{d}{dt}(l_i) = \frac{d}{dt}(\sqrt{x_i^2 + y_i^2 + z_i^2}) = \frac{\frac{1}{2}(\frac{d}{dt}(x_i^2 + y_i^2 + z_i^2))}{l_i}.$$

Notando che $z_{A,i} = z_{B,i} = 0$ per tutti gli attuatori,e che $\dot{x}_{A,i} = \dot{x}_{B,i} = \dot{y}_{A,i} = \dot{y}_{B,i} = \dot{z}_{A,i} = \dot{z}_{B,i} = 0$ si ottiene per il numeratore il seguente valore:

$$\frac{d}{dt}(x_i^2 + y_i^2 + z_i^2) =$$
$$2[\dot{x}_{AB}(x_{AB}+x_{B,i}-\psi y_{B,i}-x_{A,i})+\dot{y}_{AB}(y_{AB}+(\psi+\phi\theta)x_{A,i}+y_{B,i}-y_{A,i})+\dot{z}_{AB}(z_{AB}+(\phi\psi-\theta)x_{B,i}+(\phi+\psi\theta)y_{B,i})+\dot{\psi}(-y_{B,i}(x_{AB}+x_{B,i}-\psi y_{B,i}-x_{A,i})+x_{A,i}(y_{AB}+(\psi+\phi\theta)x_{A,i}+y_{B,i}-y_{A,i})+(\phi x_{B,i}+\theta y_{B,i})(z_{AB}+(\phi\psi-\theta)x_{B,i}+(\phi+\psi\theta)y_{B,i}))+\dot{\phi}(\theta x_{A,i}(y_{AB}+(\psi+\phi\theta)x_{A,i}+y_{B,i}-y_{A,i})+(\psi x_{B,i}+y_{B,i})(z_{AB}+(\phi\psi-\theta)x_{B,i}+(\phi+\psi\theta)y_{B,i}))+\dot{\theta}(\phi x_{A,i}(y_{AB}+(\psi+\phi\theta)x_{A,i}+y_{B,i}-y_{A,i})+(-x_{B,i}+\psi y_{B,i})(z_{AB}+(\phi\psi-\theta)x_{B,i}+(\phi+\psi\theta)y_{B,i}))].$$

Definendo le coordinate del vettore $R_{AB} = [x_{AB}\, y_{AB}\, z_{AB}]^T$ come $[x\, y\, z]^T$ (si ricordi che lo spostamento lungo gli assi è riferito al centro della piattaforma), è possibile riscrivere l'equazione della velocità come:

$$\frac{d}{dt}(l_i) = \frac{\frac{1}{2}\frac{d}{dt}(x_i^2 + y_i^2 + z_1^2)}{l_i} = \frac{k_{A,i}\dot{x} + k_{B,i}\dot{y} + k_{C,i}\dot{z} + k_{D,i}\dot{\phi} + k_{E,i}\dot{\theta} + k_{F,i}\dot{\psi}}{l_i}, \qquad (3)$$

in cui le variabili $k_{A...F}$ sono facilmente ricavabili confrontando le ultime due equazioni scritte sopra, e sono chiaramente tempo-varianti. **Tuttavia per motivi computazionali si suppongono costanti nell'orizzonte di predizione, e vengono valutate una sola volta all'inizio dell'orizzonte.**

6.5 Modelli finali dei controllori

Il metodo per il calcolo della lunghezza degli attuatori precedentemente descritto si basa sulle conoscenze della posizione della piattaforma secondo tutti i 6 gradi di libertà.
I controllori tuttavia non dispongono di tutti i gradi di libertà disponibili, ma solo di alcuni di essi. Occorre pertanto che il movimento complessivo degli attuatori sia "scomposto" nei movimenti che interessano i gradi di libertà disponibili dai controllori, in modo che ogni controllore possa calcolare tali movimenti utilizzando solo i gradi di libertà disponibili.
Dunque in questa Sottosezione verrà spiegato come approssimare la formula (3) in modo da adattarla ad ogni controllore.
Verrà inoltre spiegato come inserire le formule così ottenute nei 4 sistemi da controllare, in modo che le lunghezze degli attuatori siano contenuti nelle uscite dei sistemi. In tal modo sarà possibile porre i vincoli direttamente su tali uscite.

6.5.1 Sistema di controllo longitudinale/*pitch*

Allo scopo di rendere il sistema più semplice possibile vengono presi in considerazione soltanto i gradi di libertà relativi alla velocità longitudinale $\dot{x}$ e alla velocità angolare di *pitch* $\dot{\theta}$ (sono gli unici gradi di libertà che il controllore può gestire):

$$\frac{d}{dt}(l_i) = \frac{k_{A,i}\dot{x} + k_{E,i}\dot{\theta}}{l_i},$$

e la lunghezza degli attuatori può venire descritta dalla seguente equazione alle differenze:

$$x_{act,i}(k+1) = x_{act,i}(k) + T_s\frac{k_{A,i}\dot{x} + k_{E,i}\dot{\theta}}{l_i},$$

in cui T_s indica il periodi di campionamento.
Si ottiene così una equazione alle differenze che descrive l'evoluzione delle lunghezze degli attuatori limitandosi ai gradi di libertà disponibili al controllore, e pertanto non corrispondenti alle reali lunghezze. Tale equazione viene inizializzata con le lunghezze reali degli attuatori misurate all'inizio dell'orizzonte di predizione. Tutte queste considerazioni valgono per le analoghe equazioni che verranno sviluppate a partire dagli altri tre controllori.
Il sistema completo, includendo la dinamica vestibolare, ha la seguente struttura:

$$x_{mpc,xy}(k+1) = A_{mpc,xy}x_{mpc,xy}(k) + B_{mpc,xy}u_{xy}(k),$$

$$y_{mpc,xy}(k+1) = C_{mpc,xy}x_{mpc,xy}(k) + D_{mpc,xy}u_{xy}(k),$$

in cui lo stato ha la seguente forma:

$$x_{mpc,xy} = [x_{S_y}\, x_{\bar{O}_x}\, \theta\, x\, \dot{x}\, l_1 \dots l_6]^T,$$

mentre le matrici sono date da:

$$A_{mpc,xy} = \begin{bmatrix} A_{vest,xy} & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & T_s\frac{k_{A,1}}{l_1} & 1 & 0 & \dots & 0 \\ 0 & T_s\frac{k_{A,2}}{l_2} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & T_s\frac{k_{A,6}}{l_6} & 0 & 0 & \dots & 1 \end{bmatrix},$$

$$B_{mpc,xy} = \begin{bmatrix} B_{vest,xy} & \\ T_s & 0 \\ 0 & T_s\frac{k_{E,1}}{l_1} \\ 0 & T_s\frac{k_{E,2}}{l_2} \\ \dots & \dots \\ 0 & T_s\frac{k_{E,6}}{l_6} \end{bmatrix},$$

$$C_{mpc,xy} = \begin{bmatrix} C_{vest,xy} & I \end{bmatrix},$$

$$D_{mpc,xy} = \begin{bmatrix} D_{vest,xy} \\ 0 \end{bmatrix}.$$

L'ingresso utilizzato è dato dall'accelerazione lungo l'asse x e dalla velocità di *roll*: $u_{xy} = [a_x\, \dot{\theta}]^T$, mentre l'uscita è data da $y_{mpc,xy} = [\hat{\dot{\theta}}\, \hat{a}_x\, \theta\, x\, \dot{x}\, \dot{\theta}\, l_1 \dots l_6]^T$.

6.5.2 Sistema di controllo laterale/*roll*

I gradi di libertà sono lo spostamento lungo l'asse y e il movimento di *roll*, cioè la rotazione lungo l'asse x.
Allo scopo di rendere il sistema più semplice possibile vengono presi in considerazione soltanto i gradi di libertà relativi alla velocità laterale $\dot{y}$ e alla velocità angolare di *roll* $\dot{\phi}$:

$$\frac{d}{dt}(l_i) = \frac{k_{B,i}\dot{y} + k_{D,i}\dot{\phi}}{l_i},$$

e la lunghezza degli attuatori può venire descritta dalla seguente equazione alle differenze:

$$x_{act,i}(k+1) = x_{act,i}(k) + T_s\frac{k_{B,i}\dot{y} + k_{D,i}\dot{\phi}}{l_i}.$$

Il sistema completo, includendo la dinamica vestibolare, ha la seguente struttura:

$$x_{mpc,yx}(k+1) = A_{mpc,yx}x_{mpc,yx}(k) + B_{mpc,yx}u_{yx}(k),$$
$$y_{mpc,yx}(k+1) = C_{mpc,yx}x_{mpc,yx}(k) + D_{mpc,yx}u_{yx}(k),$$

mentre lo stato ha la seguente forma:

$$x_{mpc,yx} = [x_{S_x}\, x_{\bar{O}_y}\, \phi\, y\, \dot{y}\, l_1 \dots l_6]^T,$$

mentre le matrici sono date da:

$$A_{mpc,yx} = \begin{bmatrix} A_{vest,yx} & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & T_s\frac{k_{B,1}}{l_1} & 1 & 0 & \dots & 0 \\ 0 & T_s\frac{k_{B,2}}{l_2} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & T_s\frac{k_{B,6}}{l_6} & 0 & 0 & \dots & 1 \end{bmatrix},$$

$$B_{mpc,yx} = \begin{bmatrix} B_{vest,xy} & \\ T_s & 0 \\ 0 & T_s\frac{k_{D,1}}{l_1} \\ 0 & T_s\frac{k_{D,2}}{l_2} \\ \dots & \dots \\ 0 & T_s\frac{k_{D,6}}{l_6} \end{bmatrix},$$

$$C_{mpc,yx} = \begin{bmatrix} C_{vest,yx} & I \end{bmatrix},$$

$$D_{mpc,yx} = \begin{bmatrix} D_{vest,yx} \\ 0 \end{bmatrix}.$$

L'ingresso utilizzato è dato dall'accelerazione lungo l'asse y e dalla velocità di *roll*: $u_{yx} = [a_y\, \dot{\phi}]^T$, mentre l'uscita è data da $y_{mpc,yx} = [\hat{\dot{\phi}}\, \hat{a}_y\, \phi\, y\, \dot{y}\, \dot{\phi}\, l_1 \dots l_6]^T$.

6.5.3 Sistema di controllo dello spostamento verticale

Il grado di libertà considerato è lo spostamento lungo l'asse z. Allo scopo di rendere il sistema più semplice possibile viene preso in considerazione soltanto il grado di libertà relativo alla velocità verticale $\dot{z}$:

$$\frac{d}{dt}(l_i) = \frac{k_{C,i}\dot{z}}{l_i},$$

e la lunghezza degli attuatori può venire descritta dalla seguente equazione alle differenze:

$$x_{act,i}(k+1) = x_{act,i}(k) + T_s\frac{k_{C,i}\dot{z}}{l_i}.$$

Il sistema completo, includendo la dinamica vestibolare, ha la seguente struttura:

$$x_{mpc,z}(k+1) = A_{mpc,z}x_{mpc,z}(k) + B_{mpc,z}u_z(k),$$
$$y_{mpc,z}(k+1) = C_{mpc,z}x_{mpc,z}(k) + D_{mpc,z}u_z(k).$$

Lo stato ha la seguente forma:

$$x_{vest,z} = [x_{O_z}\, z\, \dot{z}\, l_1\, \ldots\, l_6]^T,$$

mentre le matrici sono date da:

$$A_{mpc,z} = \begin{bmatrix} A_{vest,z} & 0 & 0 & 0 & \ldots & 0 \\ 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & T_s\frac{k_{C,1}}{l_1} & 1 & 0 & \ldots & 0 \\ 0 & T_s\frac{k_{C,2}}{l_2} & 0 & 1 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & T_s\frac{k_{C,6}}{l_6} & 0 & 0 & \ldots & 1 \end{bmatrix},$$

$$B_{mpc,z} = \begin{bmatrix} B_{vest,z} \\ T_s \\ 0 \\ 0 \\ \ldots \\ 0 \end{bmatrix},$$

$$C_{mpc,z} = \begin{bmatrix} C_{vest,z} & I \end{bmatrix},$$

$$D_{mpc,yx} = \begin{bmatrix} D_{vest,z} \\ 0 \end{bmatrix}.$$

L'ingresso è costituito dall'accelerazione lungo l'asse z: $u_z = a_z$, l'uscita è data da $y_{mpc,z} = [\hat{a}_z\, z\, \dot{z}\, l_1\, \ldots\, l_6]^T$.

6.5.4 Sistema di controllo dell'angolo di *yaw*

Il grado di libertà è la rotazione lungo l'asse z, cioè l'angolo di *yaw*. Allo scopo di rendere il sistema più semplice possibile viene preso in considerazione soltanto il grado di libertà relativo alla velocità angolare di *yaw* $\dot{\psi}$:

$$\frac{d}{dt}(l_i) = \frac{k_{F,i}\dot{\psi}}{l_i},$$

e la lunghezza degli attuatori può venire descritta dalla seguente equazione alle differenze:

$$x_{act,i}(k+1) = x_{act,i}(k) + T_s\frac{k_{F,i}\dot{\psi}}{l_i}.$$

Il sistema completo, includendo la dinamica vestibolare descritta dal sottosistema $(A_{vest,yaw}, B_{vest,yaw})$, ha la seguente struttura:

$$x_{mpc,yaw}(k+1) = A_{mpc,yaw}x_{mpc,yaw}(k) + B_{mpc,yaw}u_{yaw}(k),$$

$$y_{mpc,yaw}(k+1) = C_{mpc,yaw}x_{mpc,yaw}(k) + D_{mpc,yaw}u_{yaw}(k).$$

Lo stato ha la seguente forma:

$$x_{mpc,yaw} = [x_{S_z}\ \dot{\psi}\ l_1\ \dots\ l_6]^T,$$

mentre le matrici sono date da:

$$A_{mpc,yaw} = \begin{bmatrix} A_{vest,yaw} & 0 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & \dots & 1 \end{bmatrix},$$

$$B_{mpc,yaw} = \begin{bmatrix} B_{vest,yaw} \\ T_s\frac{k_{F,1}}{l_1} \\ T_s\frac{k_{F,2}}{l_2} \\ T_s\frac{k_{F,3}}{l_3} \\ \dots \\ T_s\frac{k_{F,6}}{l_6} \end{bmatrix},$$

$$C_{mpc,yaw} = \begin{bmatrix} C_{vest,yaw} & I \end{bmatrix},$$

$$D_{mpc,yaw} = \begin{bmatrix} D_{vest,yaw} \\ 0 \end{bmatrix}.$$

L'ingresso è costituito dalla velocità angolare di *yaw*: $u_{yaw}(k) = \dot{\psi}$, l'uscita è data da $y_{vest,yaw} = [\hat{\dot{\psi}}\ \psi\ \dot{\psi}\ l_1\ \dots\ l_6]^T$.

7 Descrizione dell'algoritmo Matlab utilizzato per la risoluzione di MC

Come già detto nell'introduzione della Sezione 6, l'algoritmo di *Motion Cueing* è stato sviluppato utilizzando Matlab, mentre i sistemi da controllare sono quelli descritti nella Sottosezione 6.5. La prima parte di questa Sezione è dedicata alla descrizione delle caratteristiche del problema MPC da risolvere, in particolare i vincoli e la funzione costo. La seconda parte è dedicata alla descrizione del codice Matlab con cui è stato realizzato l'algoritmo di MC.
Questo algoritmo viene eseguito da ogni controllore, in maniera indipendente dagli altri.

7.1 Caratteristiche del problema MPC impiegato

La funzione costo utilizzata da tutti i 4 controllori ha la seguente struttura generale, con simbologia analoga a quanto visto nella sezione relativa alla trattazione di MPC:

$$J(t) = \sum_{j=1}^{N_p} \delta(j)[\hat{y}(t+j|t) - w(t+j)]^2 + \sum_{j=1}^{N_u} \lambda(j)[\Delta u(t+j-1)]^2.$$

La finestra di predizione ha dimensione $N_p = 0.25\,s$ per il controllore x/*pitch*, mentre vale $N_p = 0.3\,s$ per gli altri controllori. Poiché il periodo di campionamento vale $T_s = 0.01\,s$, allora il primo controllore valuta nella finestra di predizione le uscite dall'istante attuale t a $t + 25\,s$, mentre gli altri controllori valutano le uscite da t a $t + 30\,s$.

Le seguenti tabelle riportano i vincoli per tutti i 4 controllori utilizzati:

CONTROLLORE X/*PITCH* (Y/*ROLL*)	**Valore inferiore**	**Valore superiore**
Angolo di *pitch* e *roll*	-20 deg	+20 deg
Posizione lungo l'asse x (asse y)	-0.8 (-0.75) m	+0.8 (0.75) m
Velocità lungo l'asse x (asse y)	-2 (-1.7) m/s	+2 (+1.7) m/s
Velocità di *pitch*(*roll*)	-130 (-135) deg/s	+130 (+135 deg/s)
Accelerazione di *pitch* (*roll*)	-2000 (-2500) deg/s^2	+2000 (+2500) deg/s^2

CONTROLLORE VERTICALE	**valore inferiore**	**Valore superiore**
Posizione lungo l'asse z	-0.22 m	+0.22 m
Velocità lungo l'asse z	-1.6 m/s	+1.6 m/s
Accelerazione lungo l'asse z	-35 m/s^2	+35 m/s^2

CONTROLLORE ANGOLO DI *YAW*	**valore inferiore**	**Valore superiore**
Angolo di *yaw*	-20 deg	+20 deg
Velocità di *yaw*	-135 deg/s	+135 deg/s
Accelerazione di *yaw*	-3000 deg/s^3	+3000 deg/s^3

I vincoli per le lunghezze degli attuatori sono uguali per tutti i controllori, cioè valgono 0.75 m attorno alla posizione a riposo di 3.2691 m, quindi le lunghezze devono stare nell'intervallo [2.5191 m; 4.0191 m].

Come già mostrato in precedenza, questo problema MPC vincolato può essere espresso in forma quadratica come (si ricordi che ora $u = [\Delta u(t)\Delta u(t+1) \dots \Delta u(t+N_u-1)]^T$):

$$\min \frac{1}{2} u^T G u + b^T u + f_0$$

Con vincoli dati da $Au \leq B$.
Questi vincoli contengono sia vincoli sull'ingresso sia vincoli sull'uscita, come visto nella

Sezione 3. Questi ultimi presentano la struttura generale del tipo
$y_{min} \leq F\bar{x}(t) + Hu \leq y_{max}$.
Ricordando la struttura delle matrici F, H, e del vettore $\bar{x}$, è facile accorgersi che $F\bar{x}(t)$ dipende dal modello, mentre Hu non vi dipende. Pertanto i vincoli sulle uscite possono essere scritti come $Hu \leq B_{uscite}$ (indicati spesso con la notazione $A_{vincolo}u \leq B_{vincolo}$), in cui il secondo membro dipende da y_{min} e y_{max} e dal modello utilizzato.

Il tutto viene risolto mediante un metodo di tipo *active set* e avvalendosi di un algoritmo sviluppato in C++ e denominato qpOASES (per approfondimenti sul suo funzionamento si rimanda a http://set.kuleuven.be/optec/Software/qpOASES-OPTEC).

8 Descrizione sintetica del codice Matlab utilizzato

8.1 File *main*

Il file principale è definito `MainCodertest_script_test_458_bit.m`. Questo file legge i dati relativi al riferimento da inseguire e ai vincoli del problema dal file `DATA458` ed esegue 500 iterazioni del problema MPC. Poiché il periodo di campionamento vale $T_s = 0.01\,s$, l'intervallo temporale considerato corrisponde a $5\,s$.

Prima delle iterazioni viene eseguito il file `GenerateInitialValueForCoder_test` che inizializza i modelli dei sistemi da controllare.
All'interno di ogni ciclo i 4 controllori agiscono indipendentemente dagli altri, come illustrato dal seguente stralcio di codice:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%
%     accoppiamento x-pitch
%%%%%%%%%%%%%%%%%%%%%%%%%%
    in_xy = [IN1_XY(i), IN2_XY(i),i];

    [out_xy, y_HP_xy, y_LP_xy, rif_tot,rifperc,Yuscite ]= funzMC_XY_debug(in_xy);

    RIF(:,i) = rif_tot(1:5,:);
    RIFP(:,i) = rifperc(1:2,:);

%%%%%%%%%%%%%%%%%%%%%%%%%%
%     accoppiamento y-roll
%%%%%%%%%%%%%%%%%%%%%%%%%%
    in_yx = [IN1_YX(i), IN2_YX(i),i];
    [out_yx, y_HP_yx, y_LP_yx, rif_tot,rif_perc  ] = funzMC_YX_debug(in_yx);

    RIF_y(:,i) = rif_tot(1:5,:);
    RIFP_y(:,i) = rif_perc(1:2,:);



% %%%%%%%%%%%%%%%%%%%%
% %     movim. yaw
% %%%%%%%%%%%%%%%%%%%%
    in_za = [IN1_ZA(i),i];
    [out_za, rif_perc,rif_tot ] = funzMC_ZA(in_za);

    RIF_yaw(:,i) = rif_tot(1:3,:);
```

```
% %%%%%%%%%%%%%%%%%
% %     movim. vert.
% %%%%%%%%%%%%%%%%%
    in_zv = [IN1_ZV(i),i];
    [out_zv,rif_perc ,rif_tot] = funzMC_ZV(in_zv);

    RIF_z(:,i) = rif_tot(1:3,:);
```

La lunghezza ("reale", non secondo i gradi di libertà dei singoli controllori) dei 6 attuatori, è contenuta nelle variabili `q1,q2,q3,q4,q5,q6` e viene calcolata dal file `inverseHex` utilizzando le informazioni relative agli angoli di *pitch*, *roll*, *yaw* e alla posizione del centro della piattaforma rispetto ai 3 assi, utilizzando la cinematica inversa.

Ad ogni iterazione occorre modificare i modelli del sistema dato che, come abbiamo visto nella Sezione 6.5.1, le lunghezze degli attuatori vengono calcolate da equazioni con coefficienti tempo varianti inserite nei modelli. A tale proposito viene invocato il file `GenerateInitialValueForCoder_test_iterative` che re-inizializza i modelli prima dell'iterazione seguente.

Le equazioni che descrivono l'evoluzione delle lunghezze degli attuatori devono venire inizializzate ad ogni iterazione, inserendo come stato iniziale la reale lunghezza degli attuatori calcolata alla fine dell'iterazione corrente. Pertanto alla fine di ogni ciclo le ultime componenti dello stato di ogni sistema (corrispondenti alle lunghezze degli attuatori) vengono aggiornate con i valori di `q1,q2,q3,q4,q5,q6`:

```
Var_xy_.x_tot_xy(13:18,1)=[q1;q2;q3;q4;q5;q6];
Var_yx_.x_tot_yx(13:18,1)=[q1;q2;q3;q4;q5;q6];
Var_zv_.x_tot_zv(8:13,1)=[q1;q2;q3;q4;q5;q6];
Var_za_.x_tot_za(6:11,1)=[q1;q2;q3;q4;q5;q6];
```

Di seguito verranno descritte brevemente le funzioni più importanti che sono state implementate.

8.2 File `GenerateInitialValueForCoder_test_iterative`

In maniera analoga a `GenerateInitialValueForCoder_test`,questo file fornisce una re-inizializzazione del problema prima della successiva iterazione, invocando la funzione `inizializzazioneGUI_Matlab_iterative`.

Poiché viene chiamata ad ogni iterazione, ha un ruolo fondamentale nei tempi di calcolo, e pertanto le operazioni svolte sono solo quelle strettamente necessarie alla re-inizializzazione dei modelli.

Infatti ad ogni ciclo vengono creati solo i nuovi modelli dei sistemi da controllare e le matrici relative ai vincoli sulle uscite (quindi quelle relative a B_{uscite}), mentre tutte le altre matrici necessarie (tra le quali H e le altre matrici che definiscono il problema quadratico) vengono calcolate una sola volta da `GenerateInitialValueForCoder_test` e riutilizzate ad ogni iterazione.

8.3 File `funzMC_XY_debug`

Questo file risolve il problema di *motion cueing* per il controllore longitudinale/*pitch*. Per gli altri tre controllori sono state implementate le funzioni `funzMC_YX_debug`, `funzMC_ZA`, `funzMC_ZV`, che presentano funzionamento analogo, e dunque non verranno trattati.
La funzione utilizza i modelli prodotti da `ModelloSistemaVestibolareXPITCH` e le matrici del problema quadratico ottenute da `GenerateInitialValueForCoder_test` e da `GenerateInitialValueForCoder_test_iterative` per poi porgerli alla funzione `qpOASES_sequence_subs2` (che implementa la funzione qpOASES), la quale restituisce l'ingresso ottimo da fornire al sistema per la risoluzione del problema, consentendo quindi di calcolare l'uscita corrispondente.

9 Risultati della simulazione

9.1 Riferimenti da inseguire

Prima di analizzare i risultati ottenuti, è opportuno illustrare i riferimenti che i controllori devono inseguire.
A tale scopo verranno presentati sia i riferimenti da filtrare sia quelli già filtrati dai sistemi vestibolari descritti nella Sezione 6.3 (questi ultimi costituiscono i segnali che i controllori MPC devono effettivamente seguire). I segnali da filtrare sono ottenuti da condizioni di test tipiche, in particolare descrivono le accelerazioni e le velocità angolari che si verificano quando il veicolo simulato descrive una traiettoria compatibile con una curva a sinistra.
Nel caso del controllore x/*pitch*, il sistema vestibolare riceve in ingresso un segnale di accelerazione lungo l'asse x e un segnale di velocità angolare di *pitch*, e da questi segnali vengono ricavate come riferimenti l'accelerazione percepita lungo x, la velocità di *pitch* percepita e l'angolo di *pitch*, come spiegato nella Sezione 6.3.1.
Di seguito verranno presentati l'accelerazione lungo l'asse x in ingresso al sistema vestibolare, l'accelerazione longitudinale filtrata, la velocità di *pitch* in ingresso al sistema vestibolare e la velocità di *pitch* filtrata:

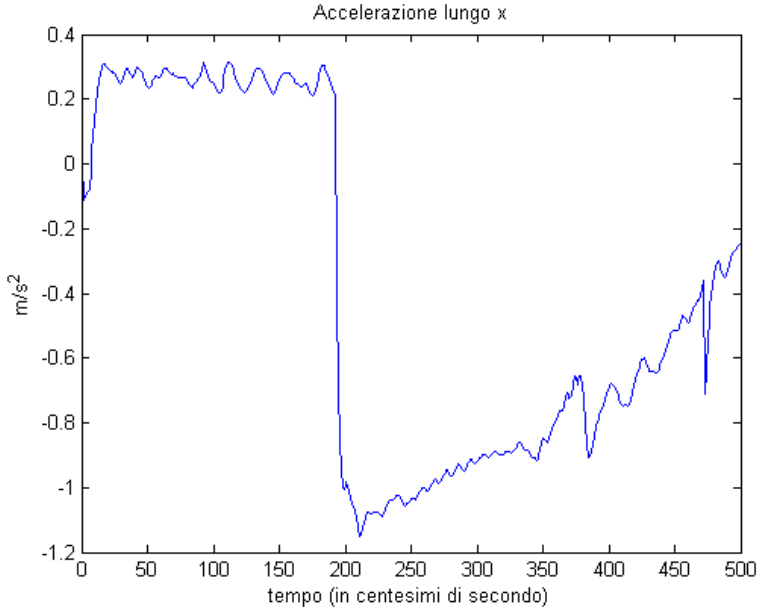

Figura 9: *segnale di accelerazione lungo l'asse x da filtrare*

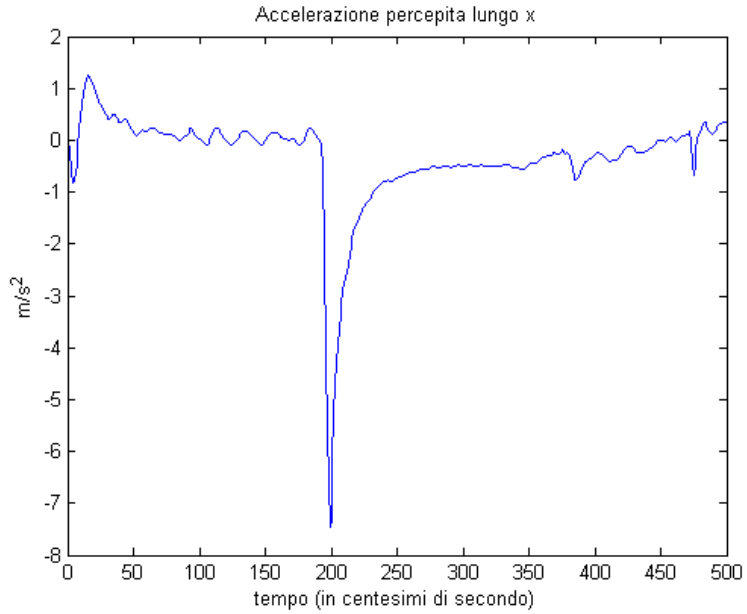

Figura 10: *segnale di riferimento per l'accelerazione percepita lungo l'asse x*

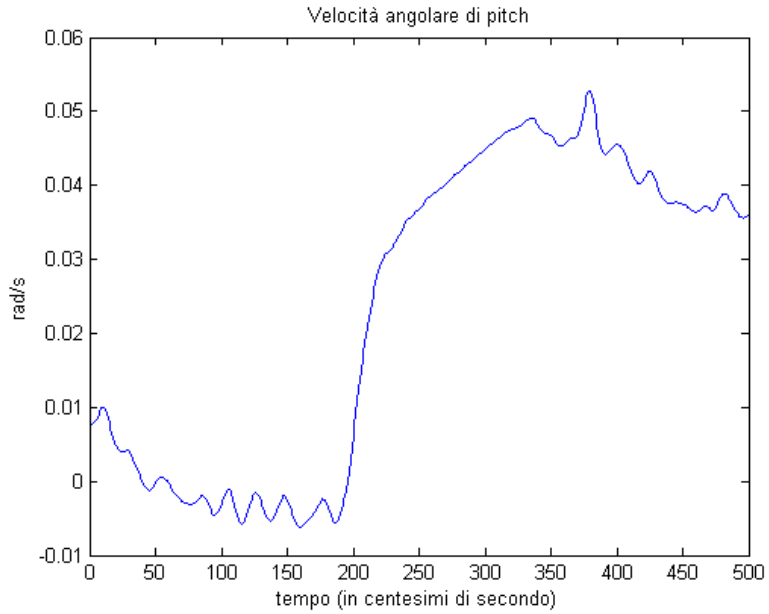

Figura 11: *segnale di velocità di pitch da filtrare*

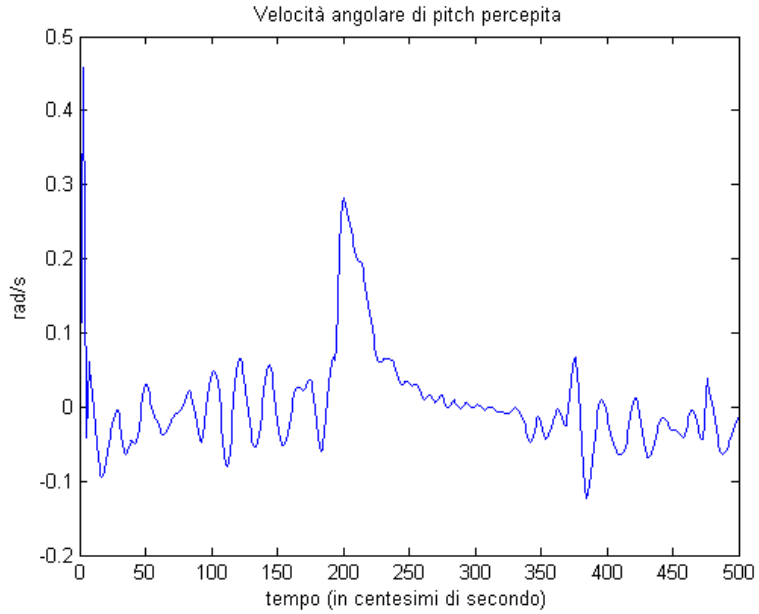

Figura 12: *segnale di riferimento per la velocità di pitch percepita*

Dalle figure si nota come i segnali di ingresso al sistema vestibolare descrivono un andamento iniziale con accelerazione longitudinale circa costante, a cui segue una fase di rapida decelerazione seguita da una fase di incremento dell'accelerazione.
Da un punto di vista del movimento angolare si assiste a una fase iniziale in cui la velocità di *pitch* tende ad assumere valori molto piccoli e negativi, seguita da una fase in cui la velocità subisce un rapido incremento e assume valori positivi (legato al fatto che durante l'incremento dell'accelerazione del veicolo si verifica una impennata verso l'alto del mezzo), per poi subire un leggero decremento.
Il sistema vestibolare calcola inoltre l'angolo di *pitch* da inseguire:

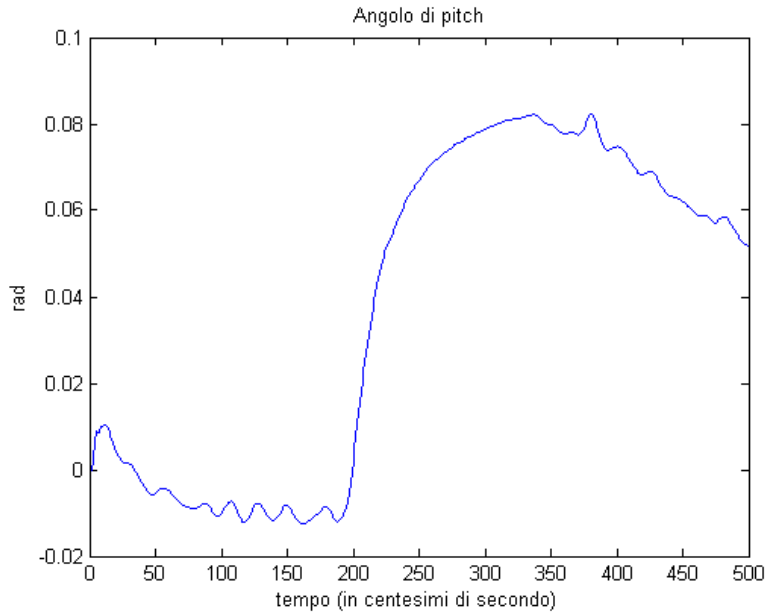

Figura 13: *segnale di riferimento per l'angolo di pitch*

Passiamo ora al controllore $y/roll$. I segnali in ingresso al sistema vestibolare e relativi riferimenti sono descritti dai seguenti grafici (per maggiori dettagli si veda la Sezione 6.3.3):

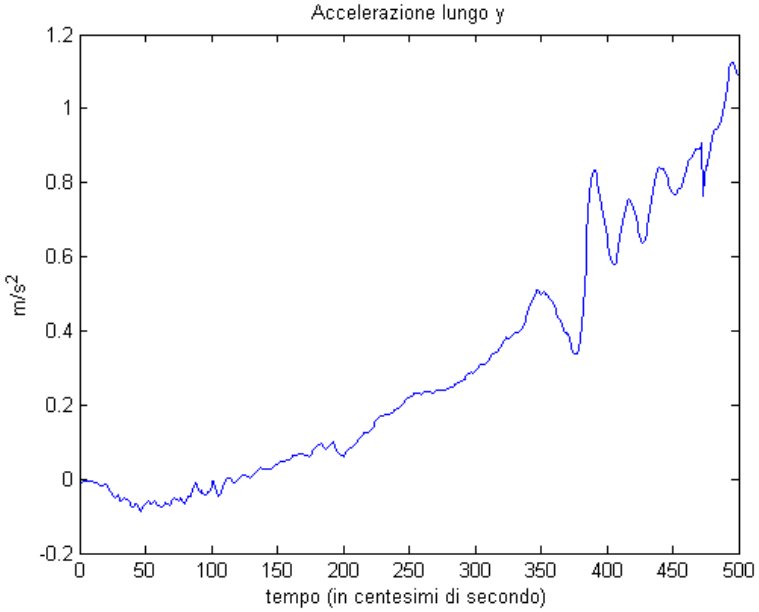

Figura 14: *segnale di accelerazione lungo l'asse y da filtrare*

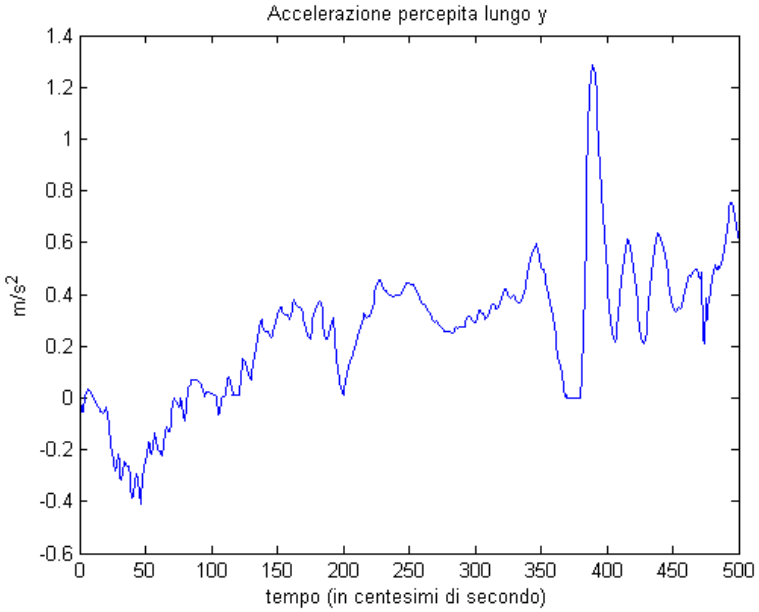

Figura 15: *segnale di riferimento per l'accelerazione percepita lungo l'asse y*

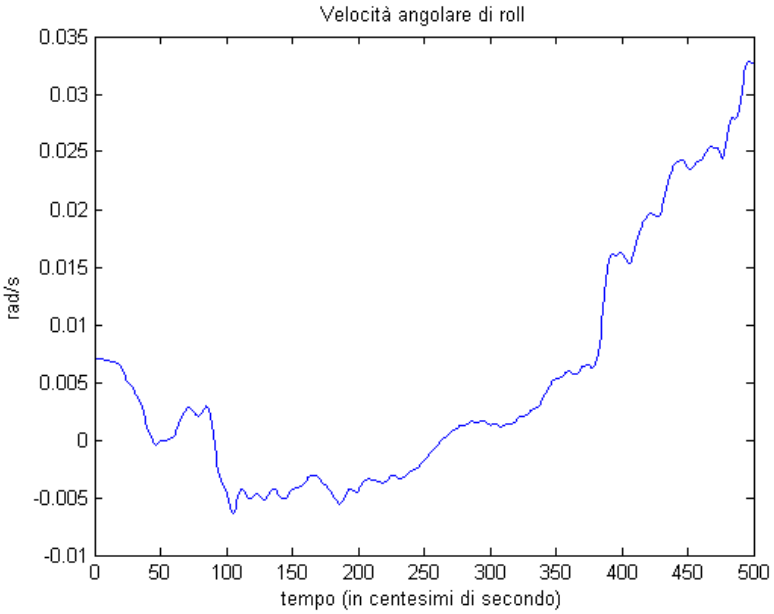

Figura 16: *segnale di velocità di roll*

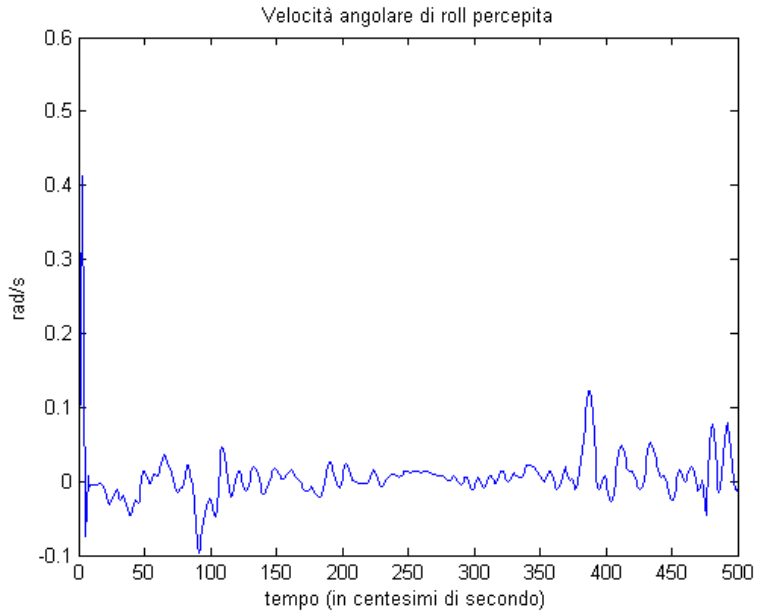

Figura 17: *segnale di riferimento per la velocità di roll percepita*

Dunque si manifesta una accelerazione lungo l'asse y grossolanamente crescente, associata ad una velocità di *roll* che inizialmente diminuisce ma poi tende ad aumentare. Questo comportamento è compatibile con quello che si verifica quando il veicolo esegue una curva a sinistra, poiché si manifesta una forza centrifuga che provoca un'accelerazione verso l'esterno mentre il veicolo tende a inclinarsi lateralmente nella prima parte della curva, per poi impennarsi leggermente verso l'alto nella seconda parte.

Il segnale per l'inseguimento dell'angolo di *roll* è invece il seguente:

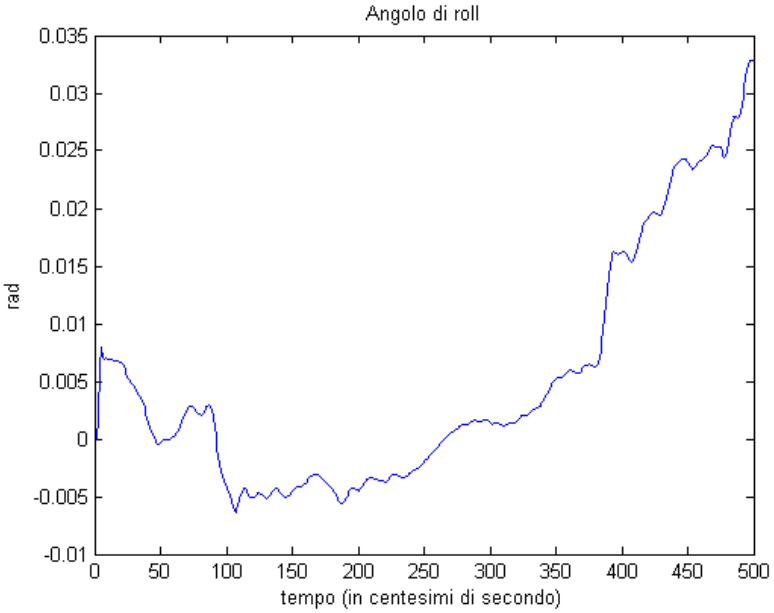

Figura 18: *segnale di riferimento per l'angolo di roll*

Si può notare che l'andamento dell'angolo di *roll* è compatibile con la tipologia di movimento appena descritto.

Passiamo ora al controllore per l'angolo di *yaw*. Il riferimento è la velocità angolare percepita di *yaw*. Dunque il sistema vestibolare di questo controllore, come spiegato nella Sezione 6.3.5, riceve in ingresso un segnale di velocità angolare di *yaw*:

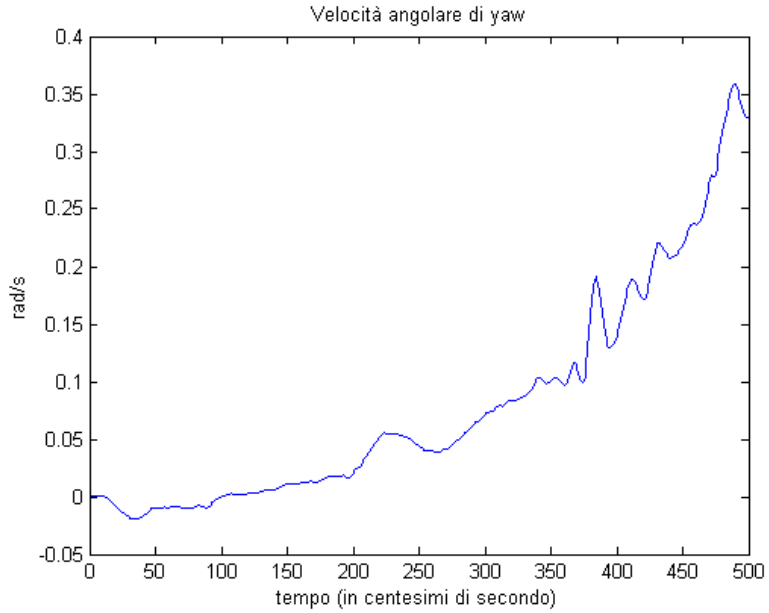

Figura 19: *segnale di velocità angolare di yaw*

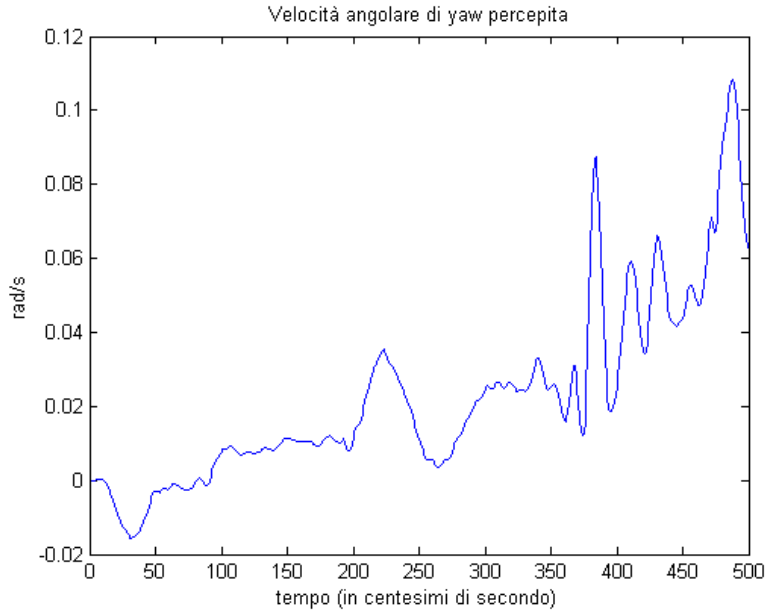

Figura 20: *segnale di riferimento per la velocità angolare di yaw percepita*

Questo segnale è compatibile con una rotazione in senso antiorario attorno all'asse z del veicolo, come effettivamente può accadere ad un veicolo mentre percorre una curva a sinistra.

Il sistema vestibolare per il controllore dell'accelerazione verticale riceve in ingresso un segnale di accelerazione lungo l'asse z (come spiegato nella Sezione 6.3.4):

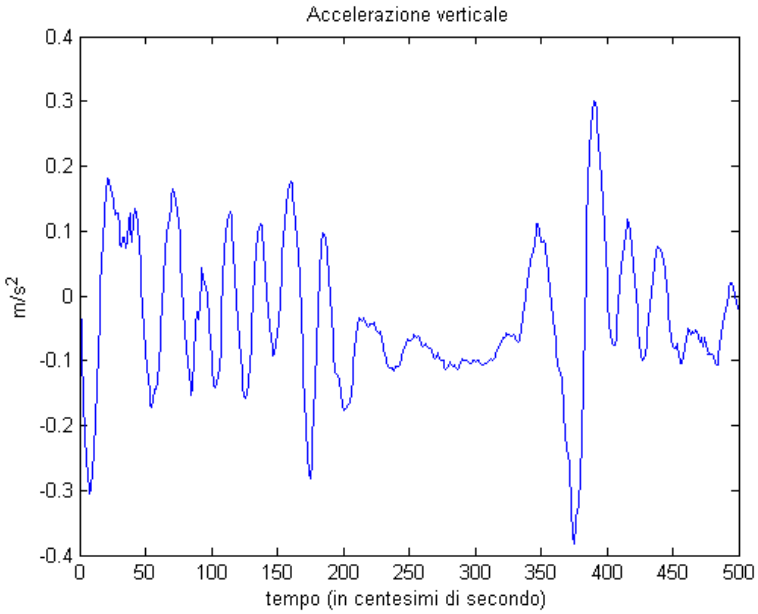

Figura 21: *segnale di accelerazione verticale*

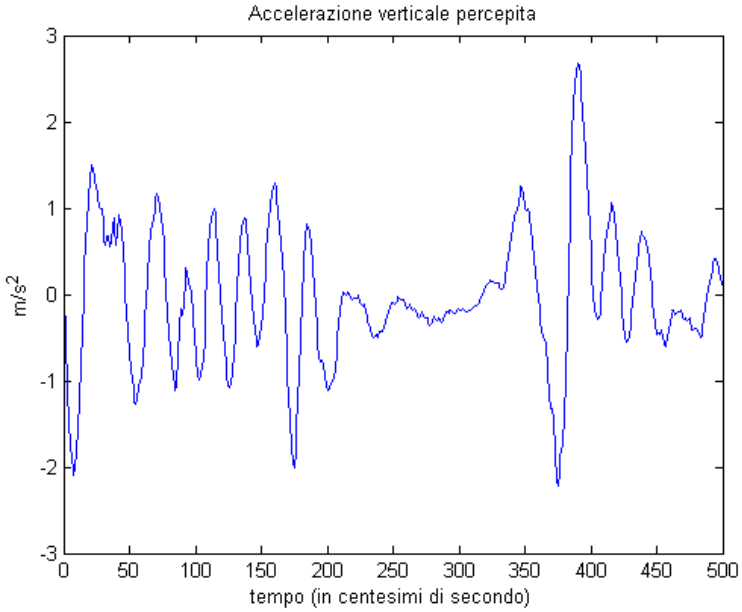

Figura 22: *segnale di riferimento per l'accelerazione verticale percepita*

In questo caso l'azione principale del filtro è quella di aumentare l'ampiezza del segnale.
Il segnale in esame descrive le vibrazioni avvertite dal veicolo. E' da notare la presenza di una zona centrale in cui il segnale si mantiene a valori prossimi allo zero, che si manifesta quando il veicolo raggiunge la parte centrale della curva.

9.2 Analisi dei risultati

Come è già stato spiegato, i 4 controllori agiscono indipendentemente, ognuno con i propri vincoli, i propri sistemi da controllare, le proprie equazioni per il calcolo delle lunghezze degli attuatori.
Tuttavia l'azione di controllo che viene esercitata sulla piattaforma è data dalla somma di queste 4 azioni di controllo indipendenti. E' lecito aspettarsi quindi che i vincoli verranno violati, dato che ogni controllore non può valutare l'azione complessiva di tutti i sistemi di controllo della piattaforma.

In effetti, analizzando l'evoluzione delle lunghezze degli attuatori (lunghezze "reali" in quanto misurate grazie alla cinematica inversa, e non stimate dai singoli controllori con i loro gradi di libertà) si nota come il vincolo inferiore (pari a 2.5191 m) viene violato dopo 305 centesimi di secondo (corrispondenti a 305 iterazioni), infatti uno degli attuatori presenta una lunghezza pari a 2.506 m, cioè inferiore al valore minimo consentito:

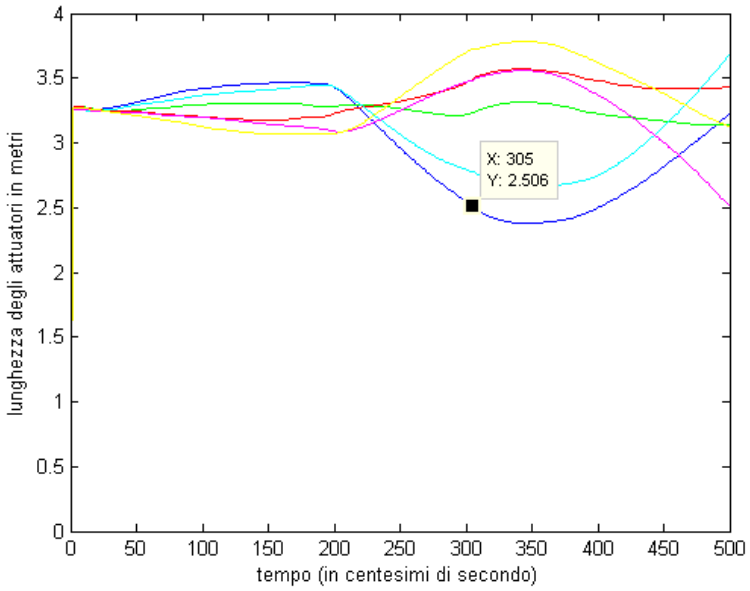

Figura 23: *evoluzione degli attuatori*

L'algoritmo risponde con un messaggio d'errore, avvertendo che i vincoli sono stati violati, come atteso.
Nelle prossime Sottosezioni verranno analizzati gli effetti di tutto ciò sulle prestazioni dei singoli controllori.

9.2.1 Risultati del controllore $x/$*pitch*

L'evoluzione delle lunghezze degli attuatori misurate secondo i gradi di libertà di questo controllore (come descritto nella Sezione 6.5.1) è molto simile a quella effettiva, e presenta una violazione di vincoli nello stesso istante.

Le prestazioni del controllore, in funzione dell'inseguimento dei segnali di riferimento filtrati, sono illustrate dai seguenti grafici:

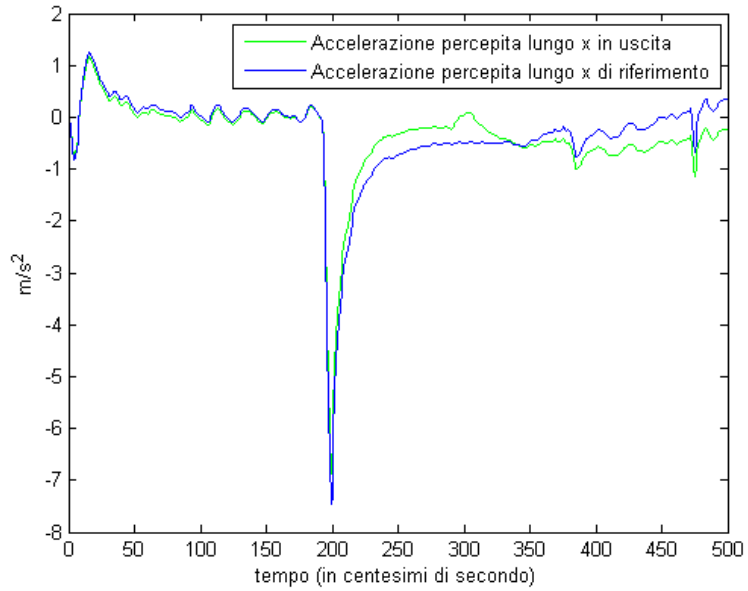

Figura 24: *inseguimento dell'accelerazione percepita lungo l'asse x per il controllore x/pitch*

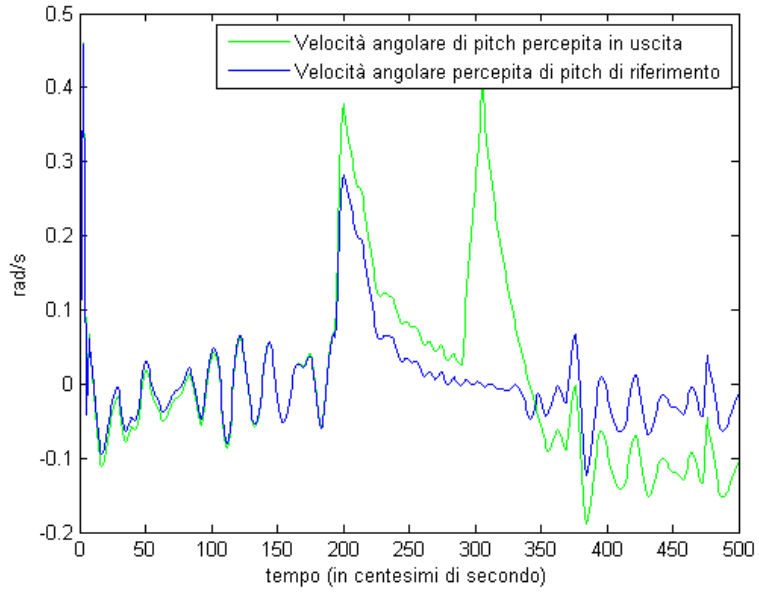

Figura 25: *inseguimento della velocità angolare percepita di pitch per il controllore x/pitch*

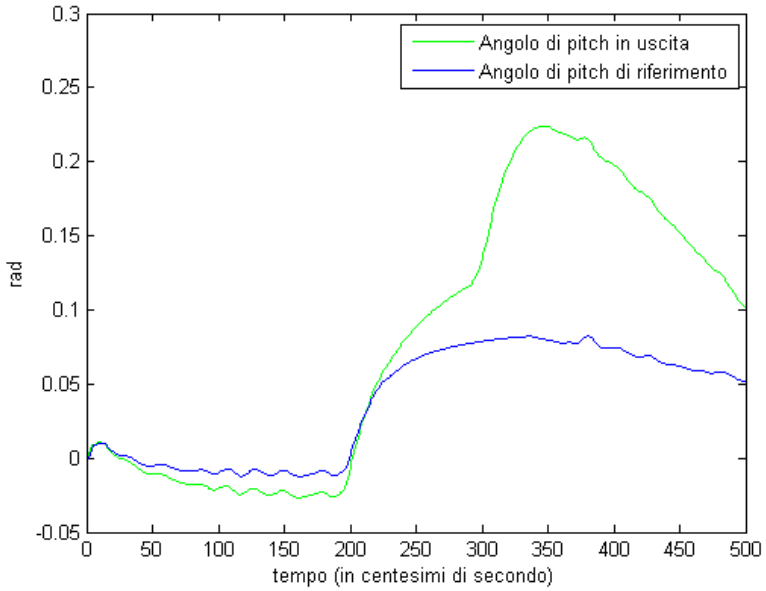

Figura 26: *inseguimento della posizione angolare di pitch per il controllore x/pitch*

Prima della violazione dei vincoli l'inseguimento dei segnali è abbastanza buono, ma dopo 305 iterazioni l'inseguimento peggiora, in particolare la velocità angolare di *pitch* presenta un vistoso picco e l'angolo fornito alla piattaforma si presenta maggiore di quanto richiesto.
In questo caso il sistema di controllo non riesce a inseguire adeguatamente il riferimento a causa della prossimità dei vincoli, e i movimenti che ne derivano causano i vistosi picchi della velocità angolare percepita di *pitch*.
Questa ipotesi è corroborata dal fatto che se i vincoli sugli attuatori vengono posti a infinito allora l'inseguimento è nettamente migliore e tali picchi spariscono.

Vengono ora riportate le altre uscite del controllore:

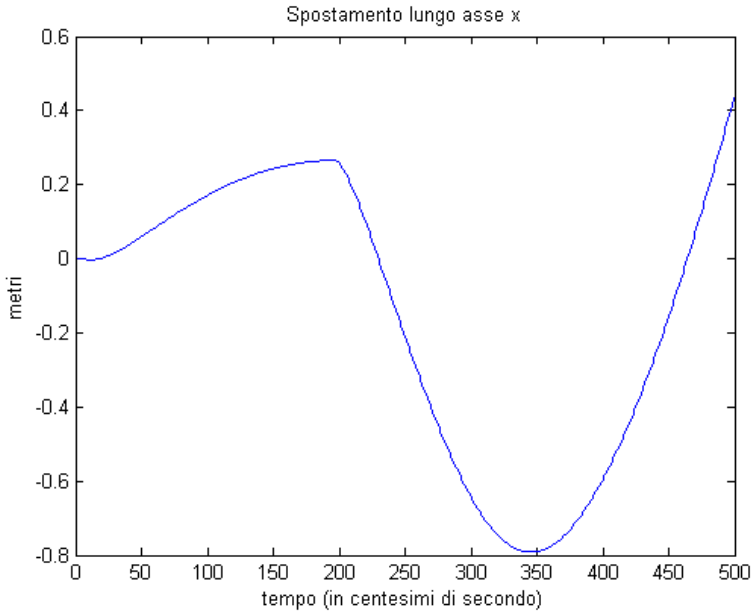

Figura 27: *Spostamento lungo l'asse x*

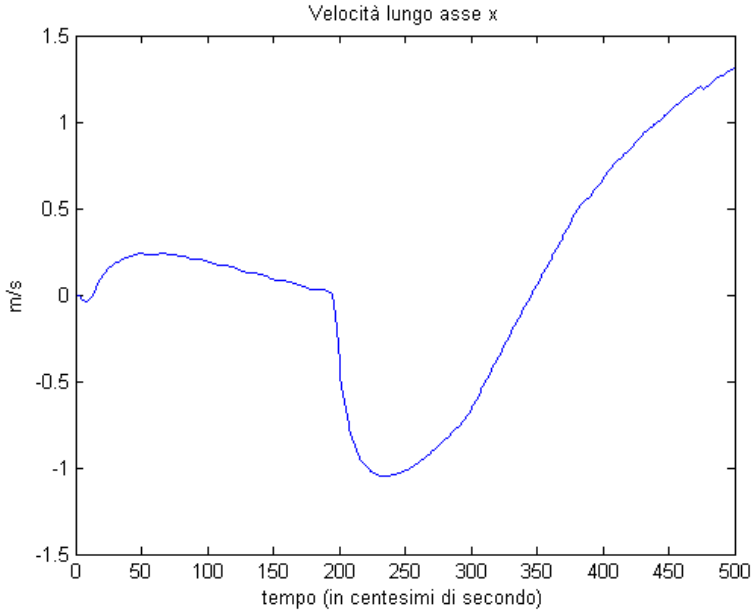

Figura 28: *Velocità lungo l'asse x*

La piattaforma quindi fino a circa 2 secondi si muove in avanti, per poi indietreggiare fino a circa 3,5 secondi e proseguire in avanti nell'ultima parte. In effetti questo movimento produce sensazioni compatibili con quelle ottenibili dal segnale di Figura 9 (cioè guidando un veicolo con tale profilo di accelerazione), costituito da una accelerazione quasi costante fino a 2 secondi, seguita da una brusca decelerazione e da un successivo aumento dell'accelerazione.

Si noti come non ci siano segni di rallentamento nelle ultime iterazioni, nonostante la lunghezza di uno degli attuatori stia per violare il limite inferiore. Il controllore anzi si

comporta come se ritenesse che tali lunghezze possano ancora rientrare nei vincoli, e si accorge solo alla fine che una delle uscite ha inaspettatamente violato i vincoli, fornendo un messaggio d'errore.

9.2.2 Risultati del controllore *y/roll*

L'evoluzione delle lunghezze degli attuatori misurate secondo i gradi di libertà di questo controllore (come descritto nella Sezione 6.5.2) è di nuovo molto simile a quella effettiva, e presenta una violazione di vincoli nello stesso istante.
Quindi anche in questo caso il controllore non si accorge della prossimità del vincolo inferiore della lunghezza degli attuatori, anzi finisce per violarlo.

L'inseguimento dei riferimenti è illustrato dai seguenti grafici:

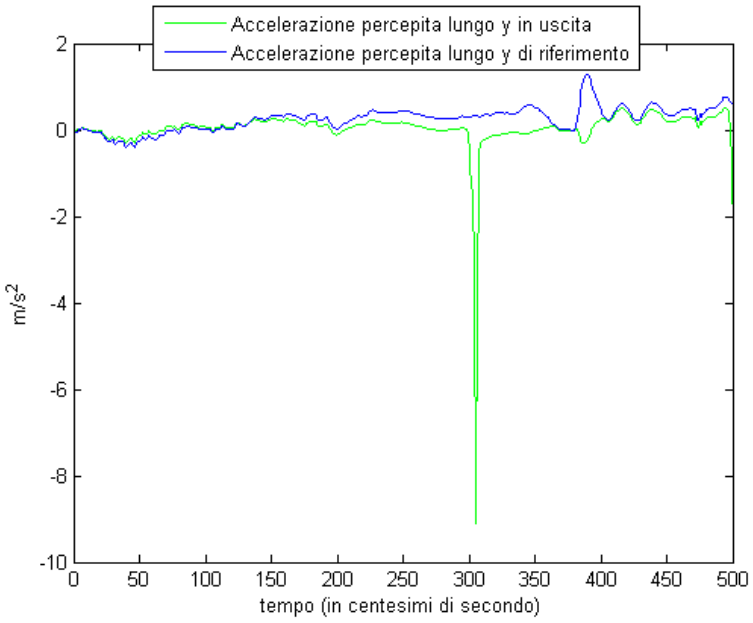

Figura 29: *inseguimento dell'accelerazione percepita lungo y per il controllore y/roll*

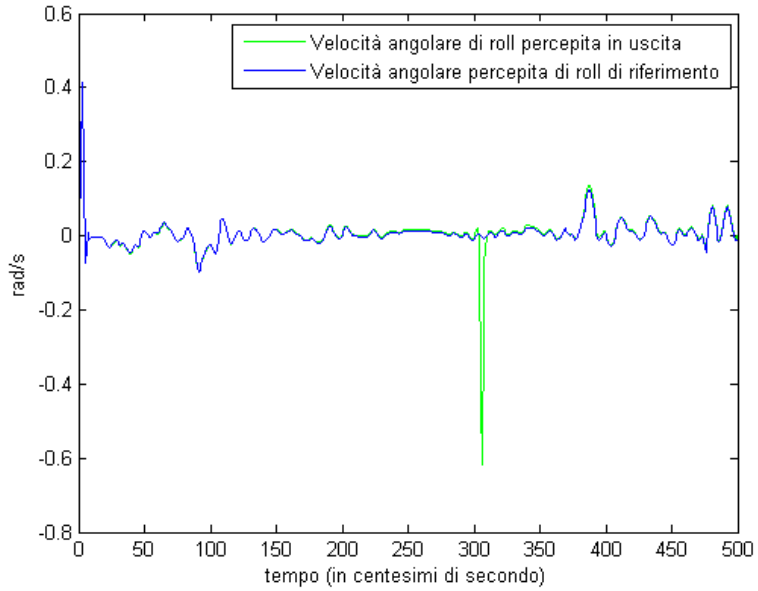

Figura 30: *inseguimento della velocità angolare percepita di roll per il controllore y/roll*

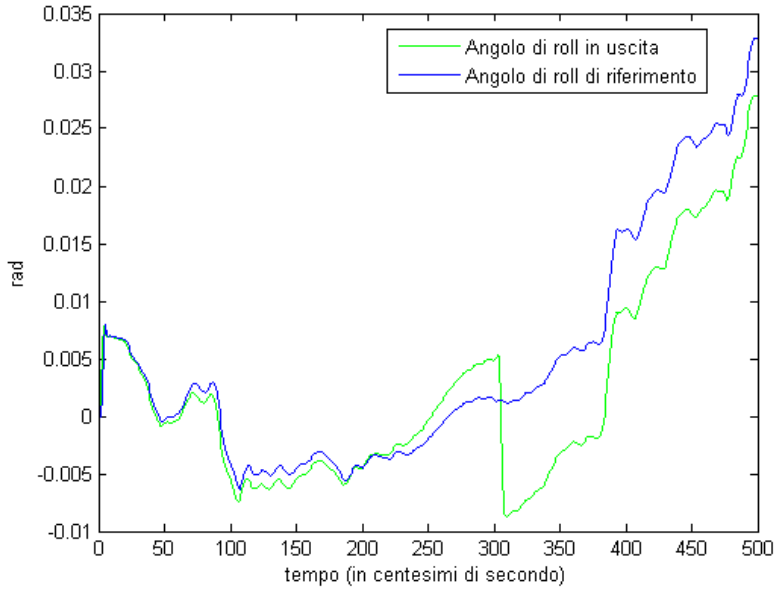

Figura 31: *inseguimento dell'angolo di roll per il controllore y/roll*

L'inseguimento dell'accelerazione lungo y e della velocità di *roll* è abbastanza buono, tuttavia in corrispondenza della violazione del vincolo è presente un brusco picco verso il basso in entrambi i casi.
Tutto ciò è dovuto agli stessi motivi prima descritti, in particolare il salto dell'angolo di *roll* provoca il vistoso picco della velocità angolare percepita.
Vengono ora riportate anche le altre uscite:

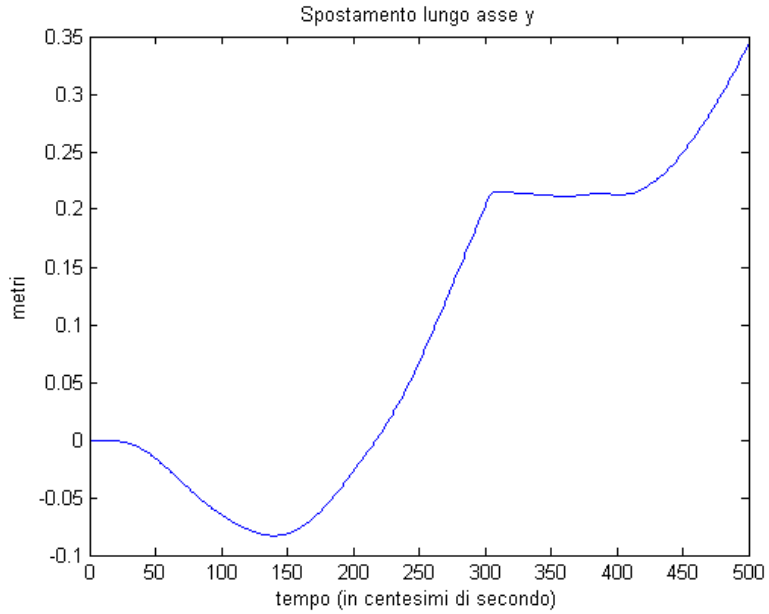

Figura 32: *spostamento lungo l'asse y*

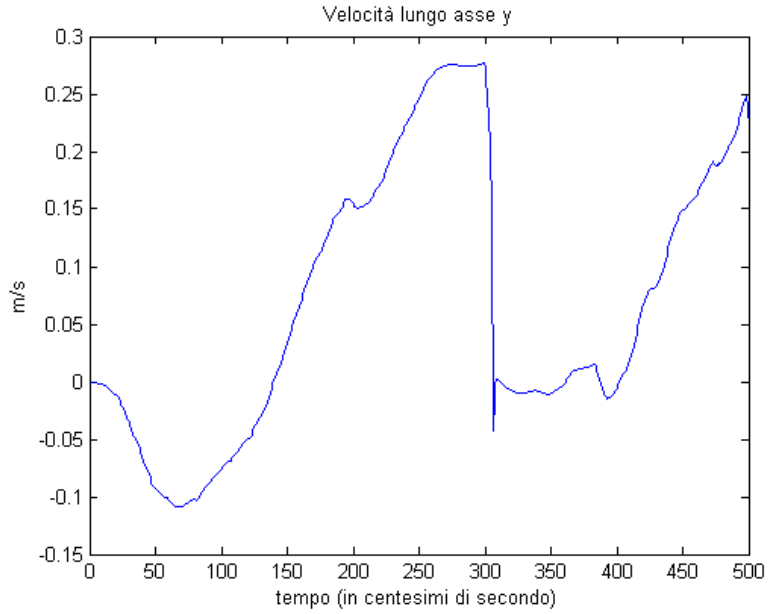

Figura 33: *velocità di spostamento lungo l'asse y*

La prossimità dei vincoli porta a un improvviso arresto nello spostamento longitudinale, e la diretta conseguenza di ciò è il picco nell'accelerazione percepita.

9.2.3 Risultati del controllore dell'angolo di *yaw*

L'evoluzione della lunghezza degli attuatori calcolata secondo i gradi di libertà del controllore dell'angolo di *yaw* è più contenuta, ad indicare come questo controllore (e i relativi gradi di libertà) non influenza in modo significativo l'andamento effettivo delle lunghezze degli attuatori:

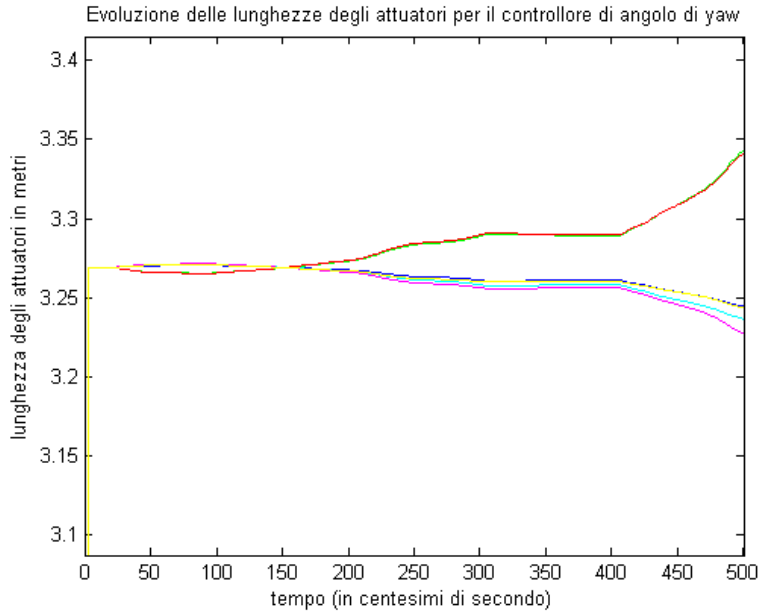

Figura 34: *evoluzione della lunghezza degli attuatori rispetto ai gradi di libertà del controllore dell'angolo di yaw*

E' da notare come in questo caso non c'è alcuna violazione dei vincoli.

Di seguito vengono riportati l'inseguimento della velocità di *yaw* e l'evoluzione di tale angolo:

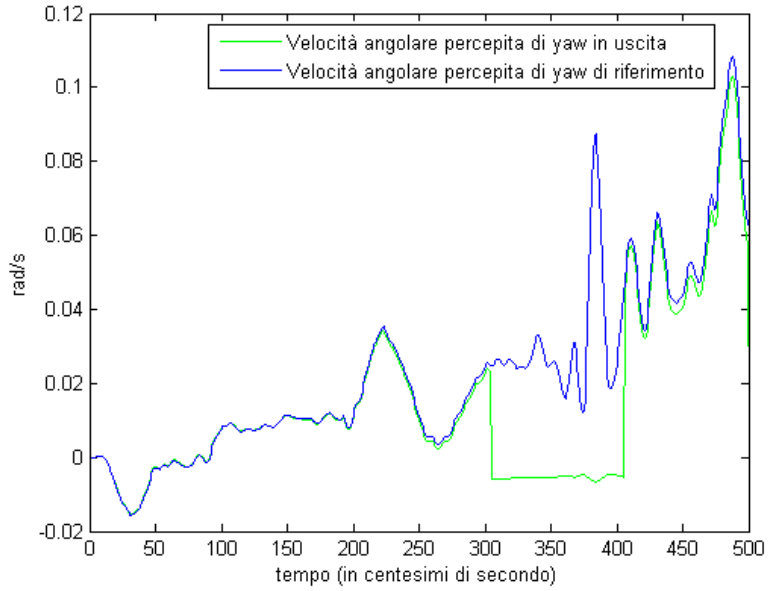

Figura 35: *inseguimento della velocità angolare percepita di yaw per il controllore dell'angolo di yaw*

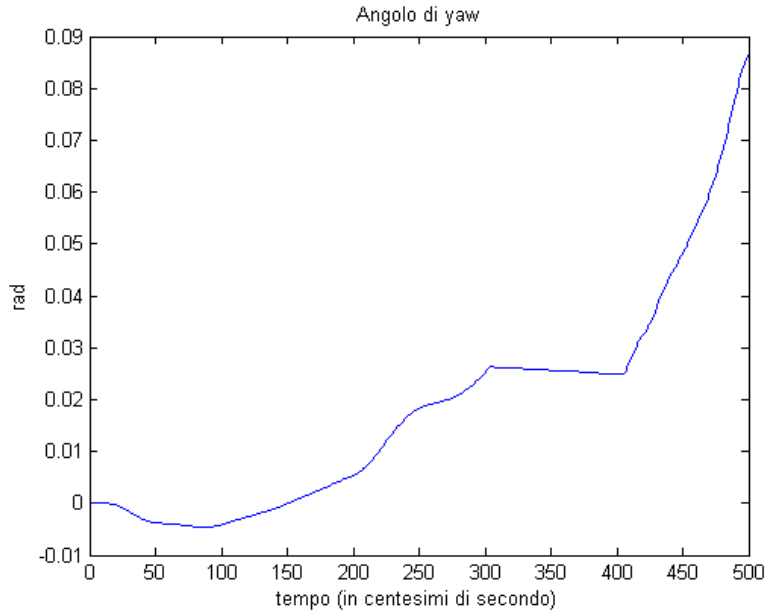

Figura 36: *evoluzione dell'angolo di yaw*

Si nota come l'angolo in questione si mantiene praticamente costante tra circa 3 e 4 secondi, di nuovo per effetto della violazione dei vincoli, provocando un vistoso scostamento dal segnale di riferimento.

9.2.4 Risultati del controllore dell'accelerazione verticale

Anche in questo caso l'evoluzione delle lunghezze degli attuatori si mantiene entro i limiti, ed anzi è molto contenuta:

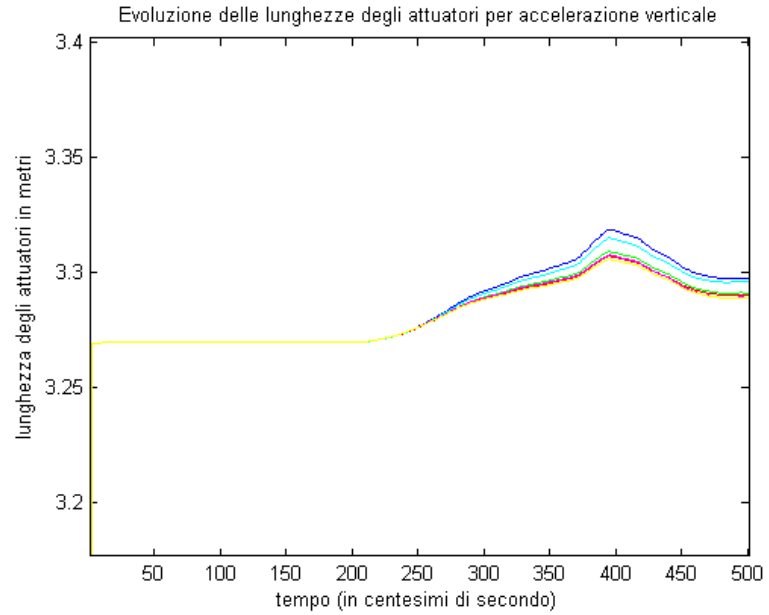

Figura 37: *evoluzione della lunghezza degli attuatori rispetto ai gradi di libertà del controllore dello spostamento verticale*

L'inseguimento del riferimento è dato dal seguente grafico:

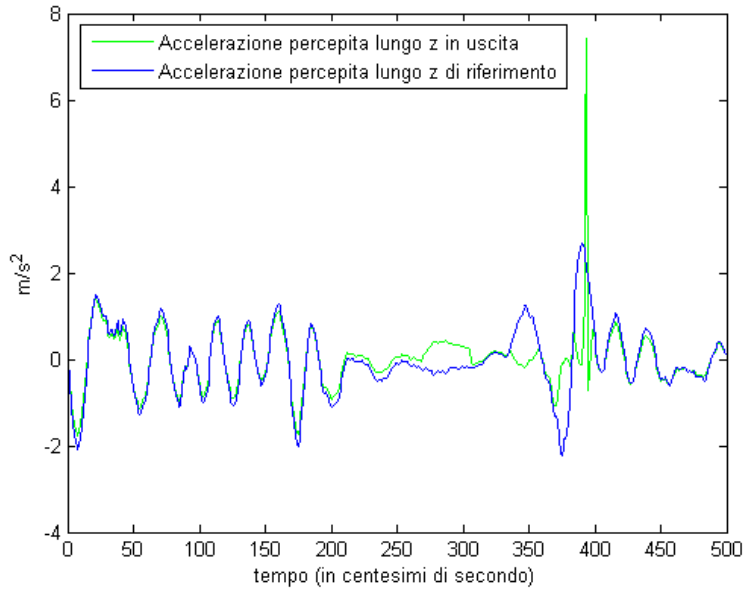

Figura 38: *inseguimento dell'accelerazione percepita lungo l'asse z per il controllore dello spostamento verticale*

In questo caso tra circa 3 e 4 secondi si verifica un forte scostamento rispetto al riferimento, con la presenza di un vistoso picco, per gli stessi motivi spiegati in precedenza, tuttavia nel seguito l'inseguimento migliora considerevolmente.
Lo spostamento e la velocità lungo l'asse z è descritto dai seguenti grafici:

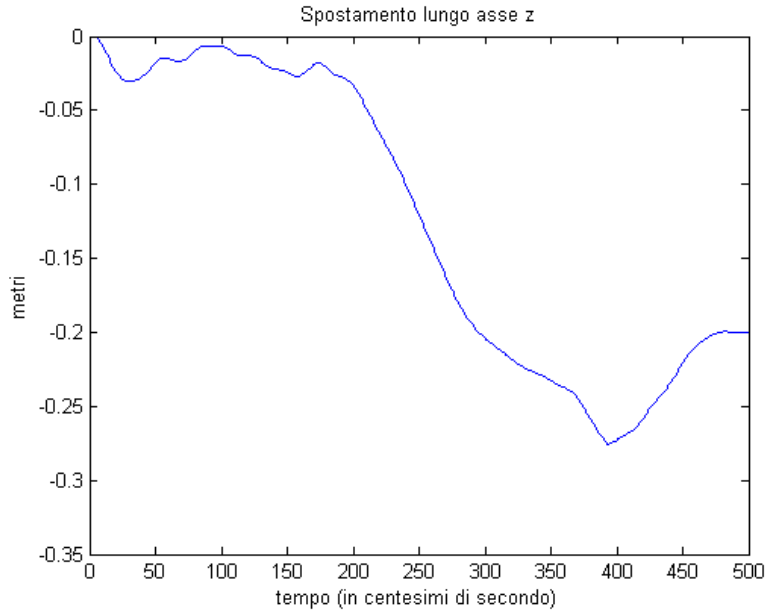

Figura 39: *spostamento verticale*

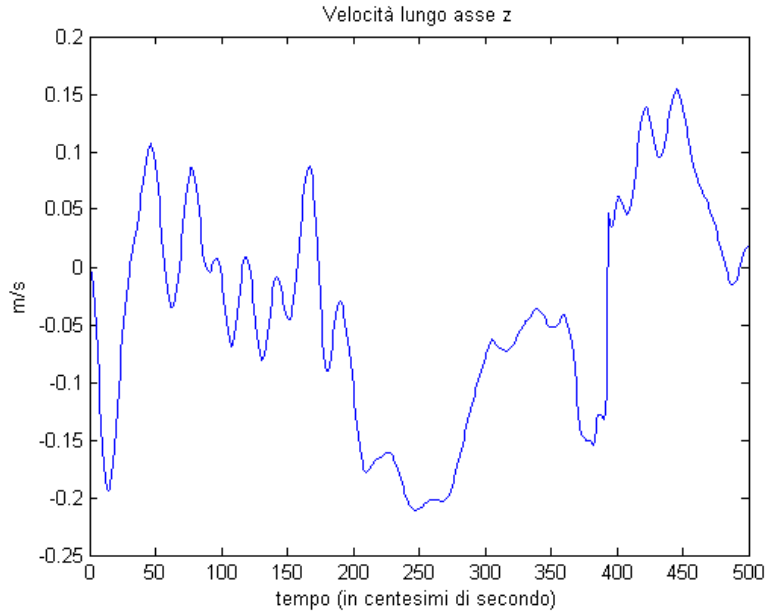

Figura 40: *velocità dello spostamento verticale*

Il brusco picco in accelerazione presente circa a 4 secondi si manifesta come un rapido spostamento verso l'alto della piattaforma.

9.3 Considerazioni sui risultati e sulla violazione del vincolo

Da quanto visto finora il vincolo viene violato dal controllore $x/pitch$ e $y/roll$. Come già spiegato la causa di ciò sta nel fatto che i 4 controllori agiscono indipendentemente tra loro.

La soluzione adottata per ovviare a questo problema si basa su una riduzione dei vincoli nominali di una certa percentuale, ottenendo dei nuovi vincoli, mentre ad ogni iterazione, corrispondente all'istante t, vengono calcolati gli stati del sistema da controllare al tempo $t+1$ in evoluzione libera, cioè senza considerare l'ingresso, e con questo valore vengono calcolate le uscite in evoluzione libera del sistema.
Utilizzando queste uscite si ottengono le lunghezze "predette" degli attuatori. Se la lunghezza massima è maggiore del nuovo vincolo superiore, allora quest'ultimo assumerà il valore di tale lunghezza. In maniera analoga si procede col nuovo vincolo inferiore.
Si otterranno cioè dei vincoli più conservativi, che verranno adattati quando serve per contenere le lunghezze degli attuatori. E' fondamentale assicurarsi che questi vincoli adattativi restino sempre all'interno della regione ammissibile di partenza (pari all'intervallo da 2.5191 m a 4.0191 m).
Nella prossima sezione verrà spiegato nel dettaglio come è stato modificato l'algoritmo e verranno presentati alcuni risultati.

10 Algoritmo modificato

10.1 Breve descrizione delle modifiche apportate

le principali modifiche apportate interessano il file principale,
`mainCodertest_ script_test_458_bit.m`. Anzitutto sono stati modificati i vincoli degli attuatori, riducendo del 10% il vincolo minimo e del 5% il vincolo massimo (questi valori percentuali sono stati scelti dopo ripetuti test al fine di ottenere le migliori prestazioni):

```
vinc_minimo=2.5191;
vinc_max=4.0191;

vinc_inf=vinc_minimo+vinc_minimo*(10/100);
vinc_sup=vinc_max-vinc_max*(5/100);
```

Le funzioni `GenerateInitialValueForCoder_test` e `GenerateInitialValueForCoder_test_iterative` sono state modificate in modo da utilizzare le variabili `vinc_inf` e `vinc_sup` come ingresso per imporre i vincoli sulla lunghezza degli attuatori.
E' stata creata con l'occasione una nuova funzione, chiamata `Correction2`:

```
[posx2,posy2,posz2,angyaw2,angpitch2,angroll2] = Correction2;
```

Essa ha il compito di utilizzare gli stati attuali dei 4 sistemi di controllo per calcolare gli stati e le uscite dei suddetti sistemi alla prossima iterazione e in evoluzione libera. Queste uscite costituiscono il comportamento "stimato" dei sistemi (si tratta di una stima perché sono in evoluzione libera), e sono il punto di partenza per calcolare la posizione lungo x, y, z e l'angolo di *pitch*, *roll*, *yaw* corrispondenti (contenuti nelle variabili di uscita `posx2`, `posy2`, `posz2`, `angpitch2`, `angroll2`, `angyaw2`).

Questi valori servono per calcolare le corrispettive lunghezze degli attuatori (`q1n`, `q2n`, `q3n`, `q4n`, `q5n`, `q6n`), utilizzando la funzione `inverseHEX`:

```
[q1n,q2n,q3n,q4n,q5n,q6n,a1n,a2n,a3n,a4n,a5n,a6n,...
b1n,b2n,b3n,b4n,b5n,b6n,Rn]=...
inverseHEX(thetap,thetag,H,D,d,posx2,posy2,posz2,angroll2,angpitch2,angyaw2);
```

Di queste lunghezze si calcola il valore minimo e massimo:

```
q_min=min([q1n;q2n;q3n;q4n;q5n;q6n]);
q_max=max([q1n;q2n;q3n;q4n;q5n;q6n]);
```

e li si confronta con i valori del vincolo inferiore e superiore. Se una delle lunghezze (o entrambe) viola uno dei vincoli (o entrambi) allora quest'ultimo viene aggiornato al valore della lunghezza:

```
if q_min<vinc_inf
    vinc_inf=q_min;
end

if q_max>vinc_sup
    vinc_sup=q_max;
end
```

Alla nuova iterazione si avranno tendenzialmente nuovi vincoli sempre più "larghi" finché non si otterrà una regione ammissibile che non verrà più violata.
Si ricorda ancora una volta che i vincoli di partenza, dati da 2.5191 m e da 4.0191 m, costituiscono i limiti fisici della lunghezza degli attuatori e perciò non devono mai venire oltrepassati.

10.2 Risultati

La prima cosa da notare è come ora le lunghezze degli attuatori rientrino nei vincoli:

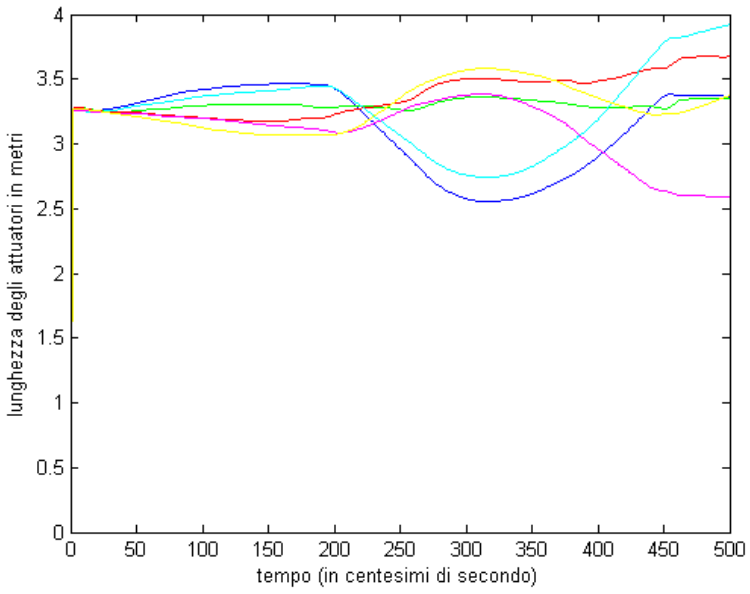

Figura 41: *evoluzione delle lunghezze degli attuatori*

Le modifiche apportate all'algoritmo hanno quindi efficacemente contrastato le violazioni della regione ammissibile.
I vincoli adattativi sviluppati vengono tuttavia violati dall'iterazione 315 alla 318 e dalla 461 alla 465, infatti i controllori continuano ad agire indipendentemente e pertanto le loro azioni di controllo si sommano portando alla violazione della regione ammissibile. Tuttavia la regione ammissibile di partenza non viene mai violata.
Adesso si analizzeranno brevemente le prestazioni dei nuovi controllori.

10.2.1 Controllore x/*pitch*

L'accelerazione percepita lungo x presenta il seguente andamento:

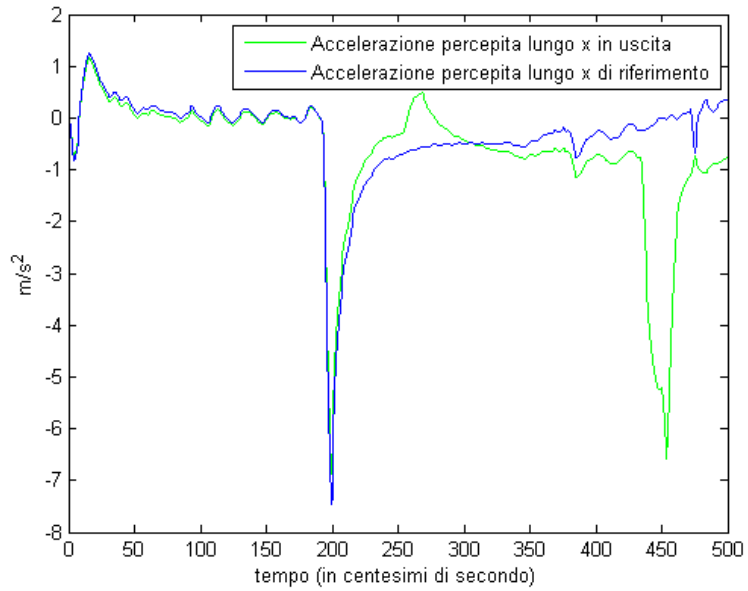

Figura 42: *accelerazione percepita lungo x*

Si nota un brusco peggioramento delle prestazioni dopo circa 250 iterazioni, confermato anche dall'evoluzione della velocità angolare percepita:

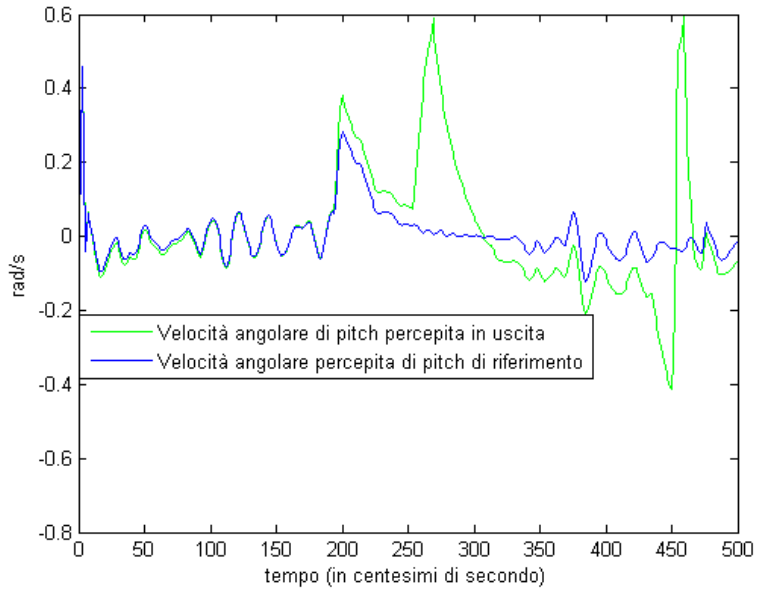

Figura 43: *velocità angolare di pitch percepita*

Tutto ciò è di nuovo provocato dalla prossimità di uno o più attuatori al vincolo di lunghezza. Occorre ricordare inoltre che la regione ammissibile è minore di quella precedente.

L'inseguimento dell'angolo di *pitch* è ovviamente legato alle scarse prestazioni dei segnali prima analizzati:

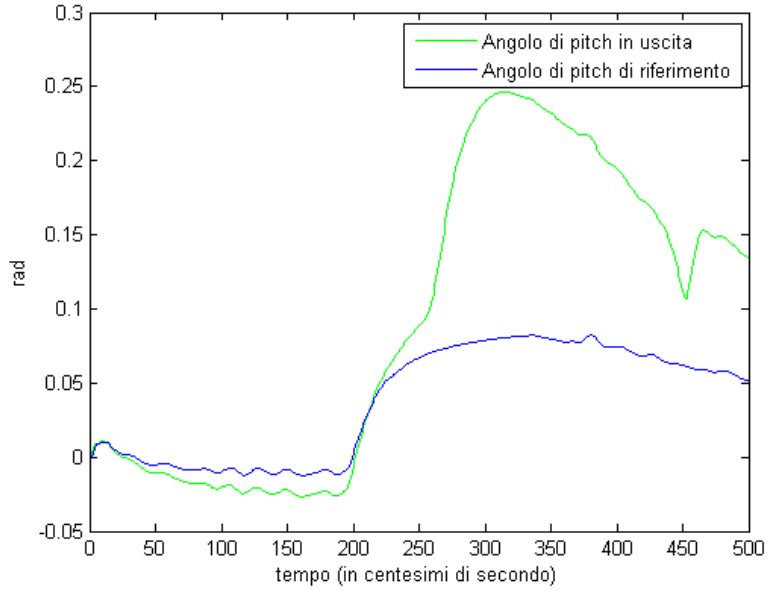

Figura 44: *inseguimento angolo di pitch*

Lo spostamento longitudinale è descritto dai seguenti grafici:

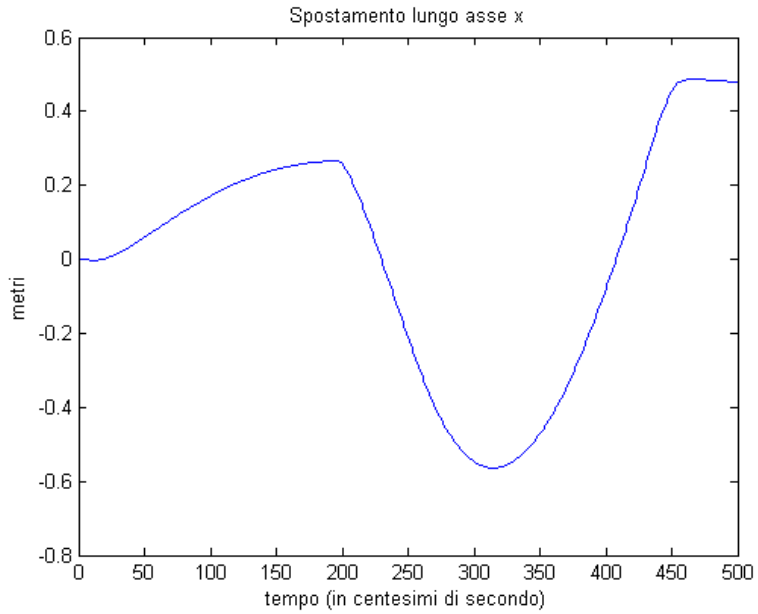

Figura 45: *spostamento longitudinale*

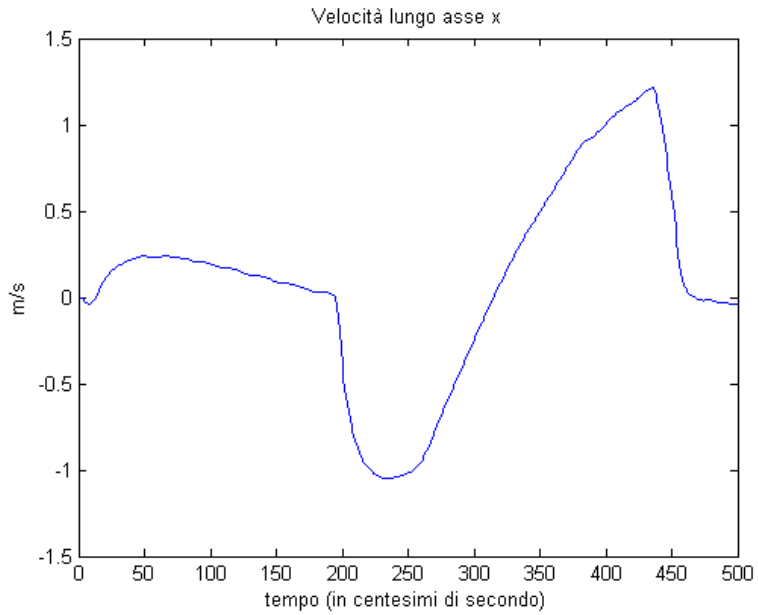

Figura 46: *velocità di spostamento longitudinale*

Lo spostamento sviluppato è compatibile col movimento richiesto dal controllore, come già spiegato nella Sezione 9.1.

10.2.2 Controllore $y/roll$

Le prestazioni in questo caso sono migliori rispetto al caso precedente:

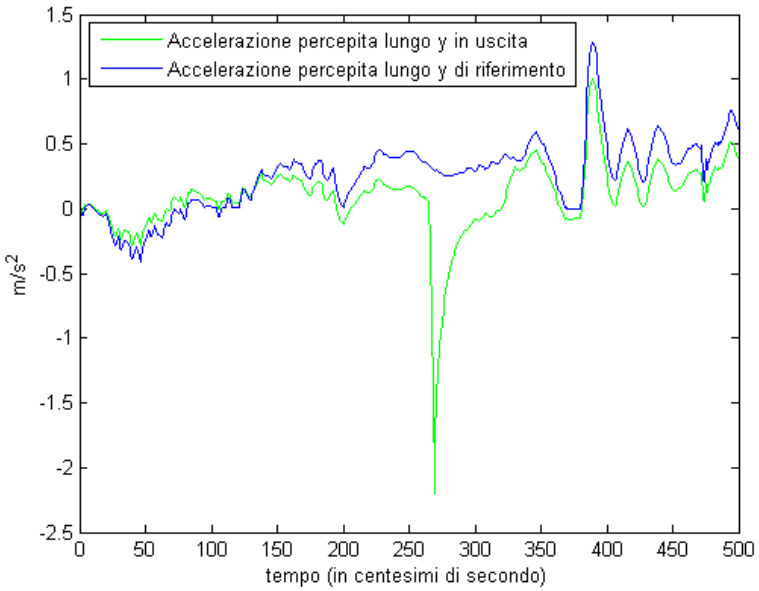

Figura 47: *accelerazione percepita lungo y*

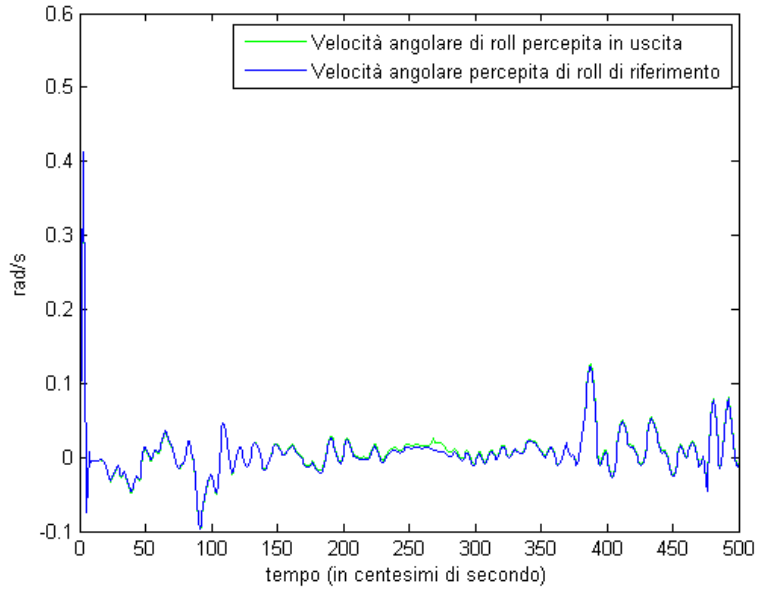

Figura 48: *velocità angolare di roll*

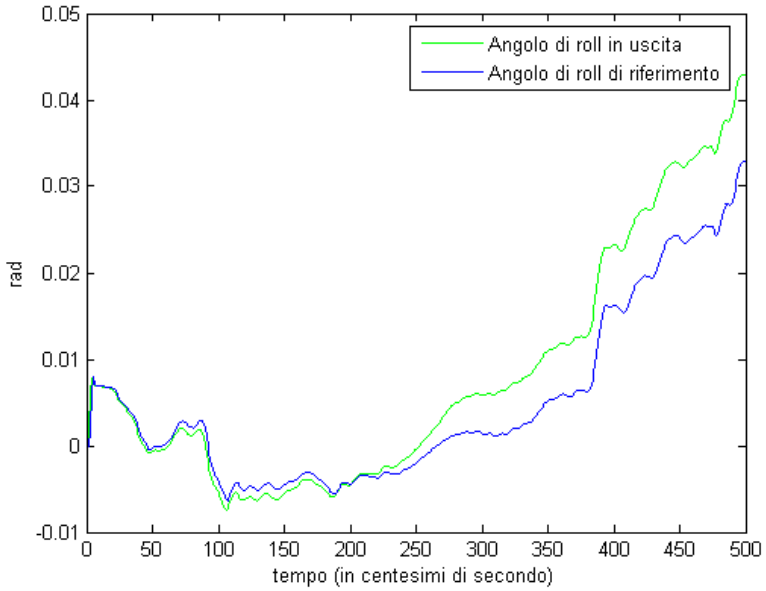

Figura 49: *angolo di roll*

Si presenta un picco di accelerazione percepita circa dopo 270 iterazioni, la cui causa è di nuovo la vicinanza degli attuatori al limite della loro lunghezza.
L'angolo di *roll* presenta uno scostamento rispetto al riferimento che si accentua dopo 2.5 secondi.

Lo spostamento laterale è descritto dai due grafici seguenti:

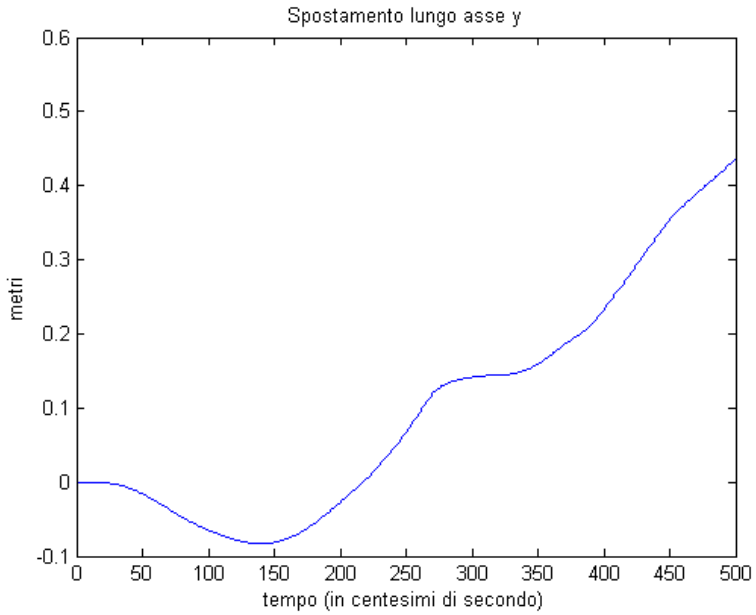

Figura 50: *spostamento laterale*

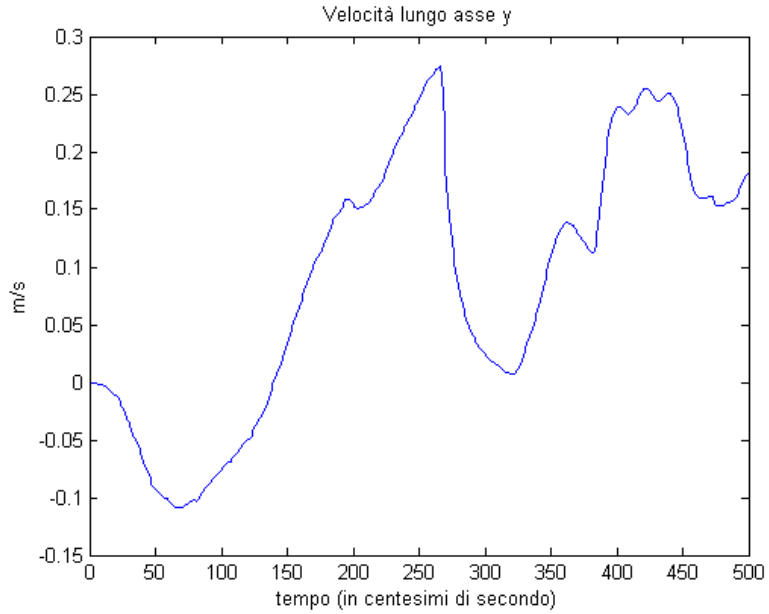

Figura 51: *velocità dello spostamento laterale*

Di nuovo l'andamento è compatibile con quanto richiesto, tuttavia tra circa 2,6 e 3,5 secondi la velocità di spostamento assume valori prossimi a zero (e la piattaforma sembra fermarsi), indicando che il movimento laterale è molto ridotto. Ciò è legato al picco negativo di accelerazione, che descrive una rapida frenata.

10.2.3 Controllori dell'angolo di *yaw* e dell'accelerazione verticale

Le prestazioni per questi due controllori sono buone:

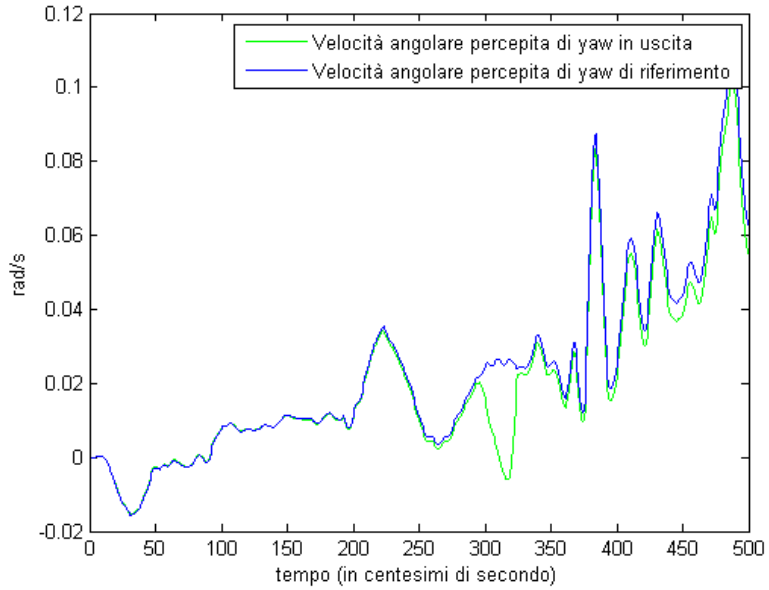

Figura 52: *velocità angolare di yaw*

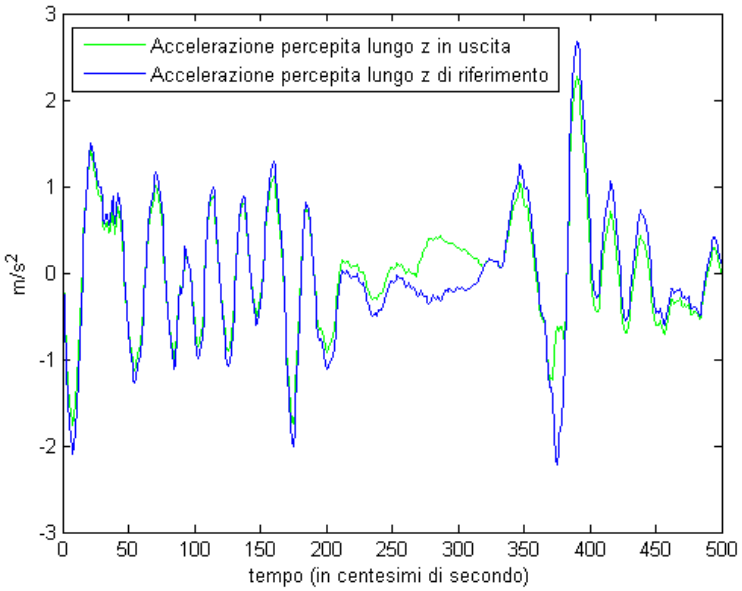

Figura 53: *accelerazione percepita lungo z*

L'andamento dell'angolo di *yaw* è il seguente:

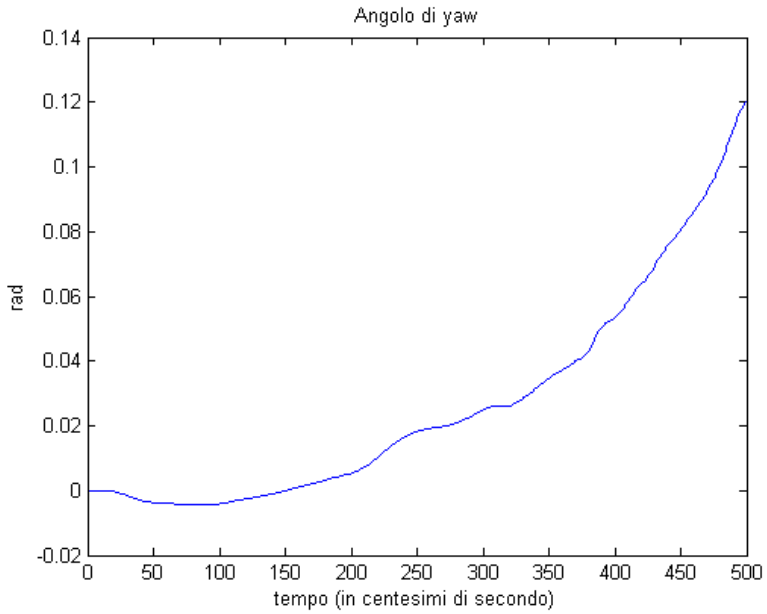

Figura 54: *evoluzione dell'angolo di yaw*

La piattaforma cioè compie una rotazione lungo l'asse z in senso antiorario.

Lo spostamento lungo l'asse z è così descritto:

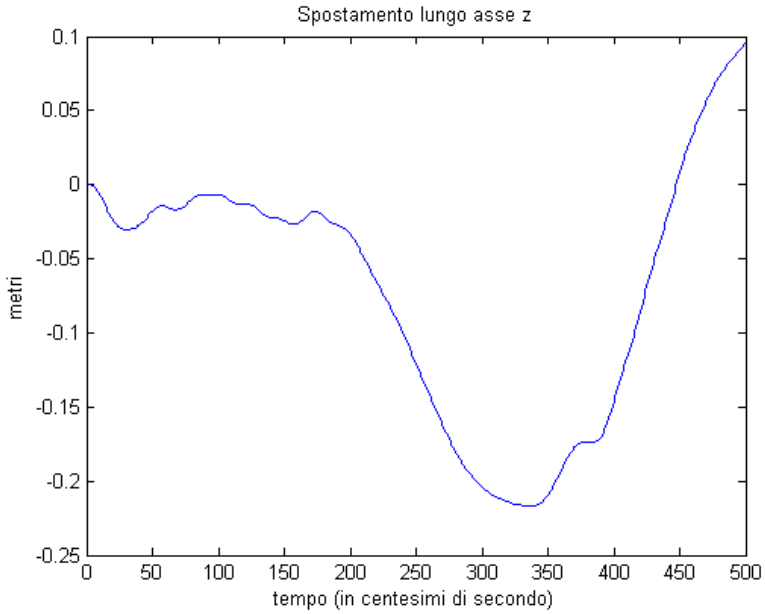

Figura 55: *spostamento verticale*

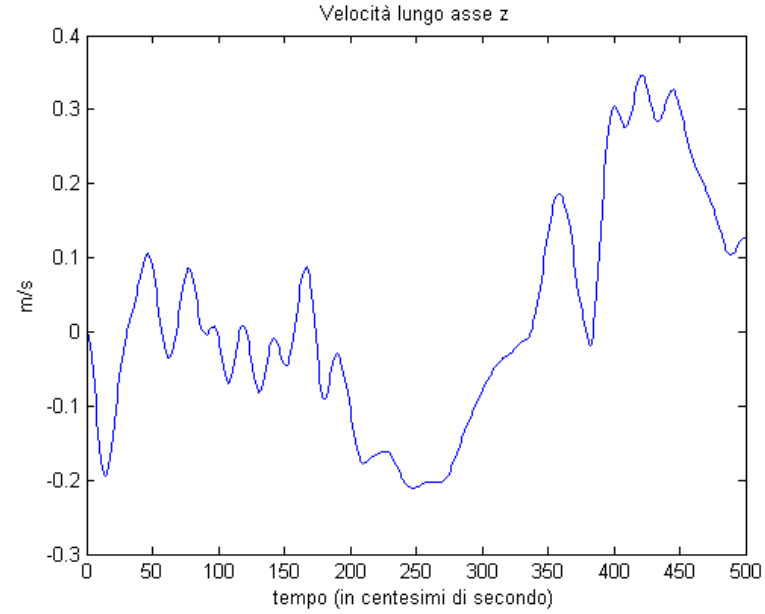

Figura 56: *velocità di spostamento verticale*

Rispetto alla Figura 40 (indicante le prestazioni dello stesso controllore con l'algoritmo originale) si nota un incremento di circa 25 cm dei valori nella seconda metà della simulazione.

11 Conclusioni e sviluppi futuri

Da quanto abbiamo visto, l'algoritmo sviluppato sul modello dell'articolo di Garrett e Best presenta alcuni **vantaggi**:

- l'inserimento dell'evoluzione della lunghezza dei vincoli nel sistema da controllare consente di utilizzare solo vincoli sull'uscita o sugli ingressi, e di ricondursi alla notazione adottata nella Sezione 3 e rendendo più efficace la risoluzione del problema di ottimizzazione.
- La separazione dei 4 controllori consente di ridurre la complessità del problema.

Tuttavia sono presenti alcuni **svantaggi**:

- ad ogni iterazione il modello del sistema da controllare viene modificato, e ciò influisce negativamente sui tempi di calcolo.
- L'indipendenza dei 4 controllori può provocare delle violazioni sui vincoli.
- La riduzione dello spazio operativo favorisce movimenti a scatti e relativi picchi in accelerazione e/o velocità angolare.

Questa strategia per la risoluzione del problema di *Motion Cueing* è stata analizzata solo parzialmente, e ci si è concentrati sul miglioramento delle prestazioni e sull'inviolabilità della regione ammissibile.
Un elemento che sarebbe opportuno analizzare è come viene modificato lo spazio operativo utilizzando i vincoli sugli attuatori. E' logico aspettarsi infatti che la presenza di questi vincoli costringa il controllore a rallentare gli spostamenti e probabilmente a non utilizzare appieno lo spazio operativo. A tale scopo sarà opportuno fornire in ingresso ad un solo sistema di controllo (ad esempio quello $x/pitch$) un riferimento in accelerazione a gradino, ponendo a zero tutti gli altri riferimenti. In tal modo si costringe il controllore ad agire solo su un grado di libertà, e si può agevolmente confrontare l'utilizzo dello spazio di lavoro rispetto a quello utilizzato da un controllore analogo ma privo di vincoli sulla lunghezza degli attuatori.
Nel seguito si combinerà il riferimento a gradino con altri riferimenti, al fine di ottenere un movimento più realistico e vedere se e come l'azione combinata degli altri controllori riduce ulteriormente lo spazio operativo effettivamente utilizzato dalla piattaforma.

Riferimenti bibliografici

[1] Nikhil J. I. Garrett, Matthew C. Best, 2013, *Model predictive driving simulator motion cueing algorithm with actuator-based constraints*, Vehicle System Dynamics: International Journal of Vehicle Mechanics and Mobility

[2] E. F. Camacho, C. Bordons (2007), *Model Predictive Control*, Springer

[3] J. M. Maciejowski (2000), *Predictive Control with Constraints*, Prentice Hall

[4] Liuping Wang (2009), *Model Predictive Control System Design and Implemetation Using Matlab*, Springer

[5] Alessandro Beghi, Mattia Bruschetta, Fabio Maran (2012), *A real time implementation of MPC based Motion Cueing strategy for driving simulators*, CDC, VI-Grade Italy e Università di Padova

[6] R. Telban, W. Wu, F. Cardullo, L. R. Center (2000), *Motion Cueing Algorithm Development: Initial Investigation and Redesign of the Algorithms*, National Aeronautics and Space Administration, Langley Research Center

[7] Ettore Fornasini (2012), *Appunti di Teoria dei Sistemi*, Edizioni Libreria Progetto Padova

[8] http://en.wikipedia.org/wiki/Model_predictive_control

[9] Riccardo Scattolin, Dipartimento di Elettronica e Informazione, Politecnico di Milanoftp://ftp.elet.polimi.it/users/Riccardo.Scattolini/MPC

[10] Bruno Daniel Correia Augusto, Rui Josè Leal Loureiro (2009), *Motion Cueing in the Chalmers Driving Simulator: A Model Predictive Control Approach*, Department of Signals and Systems, Division of Automatic Control, Automation and Mechatronics, Chalmers University Of Technology, Göteborg, Svezia

[11] S. Joe Quin, Thomas A. Badgwell, 2002, *A survey of industrial model predictive control technology*, Control Engineering Practice

[12] Manfred Morari, Jay H. Lee, 1997, *Model Predictive Control: Past, Present and Future*, Joint 6th International Symposium on Process Systems Engineering, Trondheim, Norvegia

[13] Elizabeth Wong, 2011, *Active-Set Methods for Quadratic Programming*, University of California, San Diego