**Introduzione al problema e inizializzazione dei parametri**

Nel seguente progetto di Controllo dei Sistemi Incerti ci proponiamo lo studio tramite alcuni strumenti di sintesi di un controllore, del sistema costituito da un convertiplano. In particolare, utilizzeremo il controllore **LQG**, **Mix-Sensitivity**, **H-Infinity** e come ultimo un controllore ***µ-Synthesis*** con la ***DK-Iteration***. Andremo poi a studiare tali controllori sfruttando lo strumento della ***µ-Analysis***. Per simulare e testare i controllori precedentemente esposti ci siamo avvalsi dell’ambiente Matlab Simulink®.

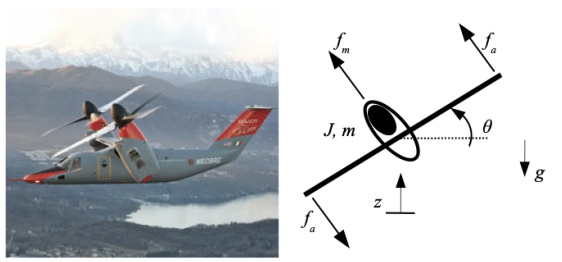


Figura 1.1 – Sistema Convertiplano

* 1. **Parametri Fisici**

Per impostare i parametri fisici nominali del sistema abbiamo tenuto conto della ragionevolezza dei valori per il sistema fisico reale in studio. Di seguito riportiamo l’elenco di questi con le relative unità di misura nel Sistema Internazionale:

**Convertiplano**

* Massa della struttura: ………………………………………………….
* Momento di inerzia associato: ……………………………………….
* Coefficiente di attrito viscoso di **traslazione** con l’aria: …….
* Coefficiente di attrito viscoso di **rotazione** con l’aria: ……….
* Apertura alare del convertiplano: …………………………………..
* Accelerazione di gravità: ………………………………………………

**Attuatori** (modellati come funzioni di trasferimento)

* Guadagno statico motori: …………………………………………..

………………………………………….

* Costante di tempo poli: ……………………………………………..

………………………………………………

* Costante di ritardo: ……………………………………………………

……………………………………………………

* 1. **Modello dinamico**

Le equazioni differenziali sotto riportate rappresentano il modello dinamico del sistema fisico, ovvero descrivono l’evoluzione degli stati del convertiplano così definiti:

* Altitudine del convertiplano:
* Posizione angolare rispetto alla direzione longitudinale:
* Velocità verticale del convertiplano:
* Velocità angolare del convertiplano:

Con la seguente dinamica:

In queste equazioni notiamo la presenza dei parametri e che rappresentano le forze che vengono impresse dagli attuatori del convertiplano, nonchè il nostro ingresso di controllo.

Invece le funzioni di trasferimento che modellano i motori sono date da:

* 1. **Modellazione e analisi sistema**

Linearizzando il sistema intorno al punto di equilibrio , cioè il nostro **obiettivo di controllo**, abbiamo verificato che fosse **completamente raggiungibile**:

Da cui:

Dove lo stato è dato da:

Quindi abbiamo scelto l’uscita del sistema come gli stati e , così da rendere il sistema linearizzato anche **completamente osservabile**:

Perciò possiamo effettuare un **controllo a zero** del nostro sistema, cioè fare in modo che il convertiplano rimanga ad una altitudine costante () e con inclinazione costante ().

**Progettazione e modellazione su MATLAB Simulink®**

La dinamica del sistema è stata modellata su MATLAB Simulink® a partire dalle equazioni proposte, tramite schemi a blocchi e Matlab Function. I dati temporali relativi alla simulazione sono i seguenti:

Per la modellazione su Simulink® abbiamo diviso il sistema fisico in sotto-sistemi (subsystems), ciascuno rappresentante parti specifiche della simulazione, come possiamo vedere di seguito:

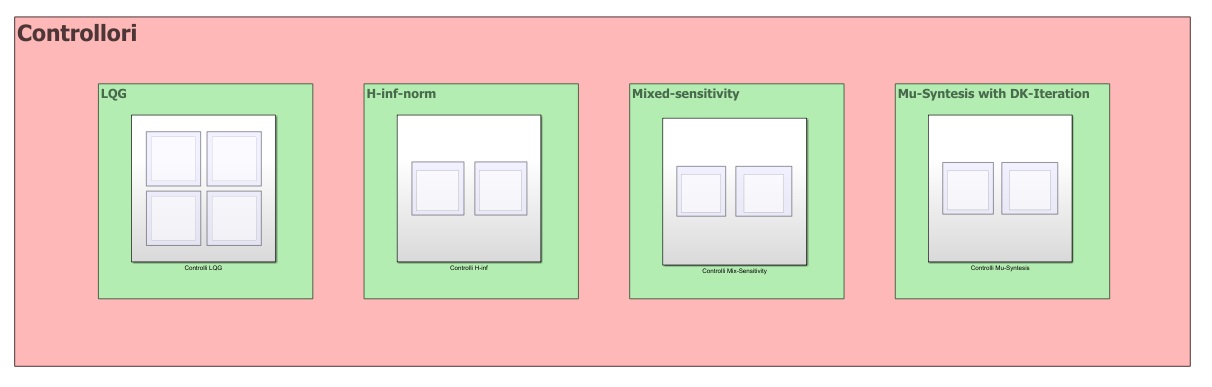


Figura 2.1 - Schema a blocchi del modello Simulink®

Di seguito approfondiremo ciascun blocco, osservando la costruzione e le relative osservazioni/conclusioni di ciascuno.

**2.1 Controllore LQG**

Il primo passo effettuato per andare a ricavare il controllore LQG, è stato **linearizzare** il sistema intorno al **punto di equilibrio** su cui vogliamo andare a **controllarlo** (come già visto precedentemente nell’analisi di Raggiungibilità e Osservabilità).

Fatto ciò, abbiamo unito gli attuatori al sistema dinamico del convertiplano così da avere un sistema unico:

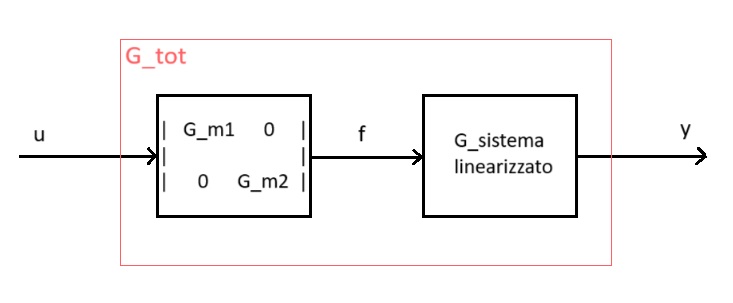


Figura 2.2 – Composizione sistema dinamico con sistema degli attuatori (composizione serie)

Quindi abbiamo cercato il sistema sfruttando la proprietà di **composizione** di **sistemi in serie** nella loro forma di stato, così da capire quali stati dover pesare per poi applicare il comando ***lqg()*** su MatLab.

Dunque, il codice usato per ricavare in forma di stato è il seguente:

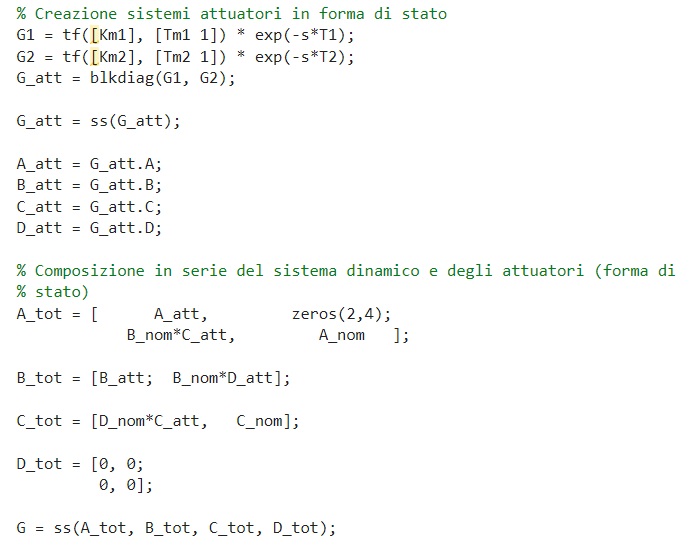


Figura 2.3 – Codice per la composizione in serie di sistemi

Fatto ciò, abbiamo definito la **matrice di peso** (LQR) e la **matrice di covarianza** dei rumori bianchi (KF) così da usarle per il codice *lqg()* che ci restituisce i guadagni per l’LQR e per il KF:

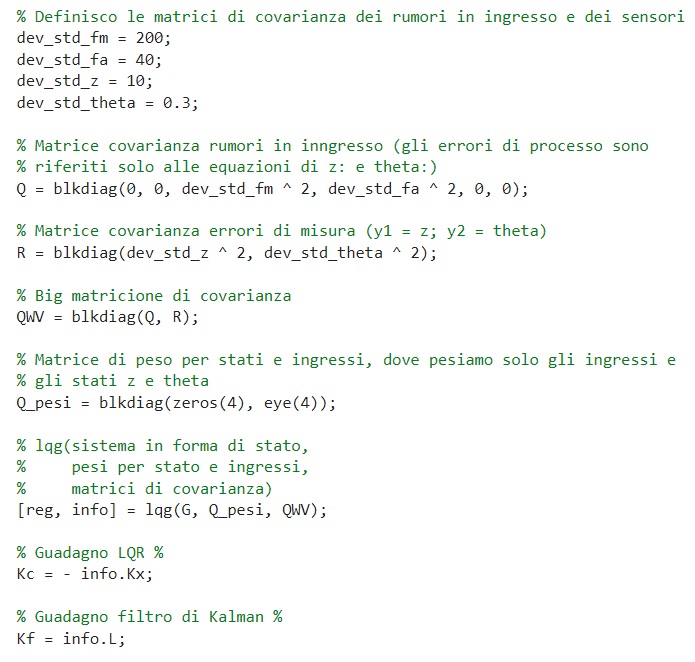


Figura 2.4 – Codice per calcolo guadagni costanti con lqg()

Infine, per fare un **controllore LQG** con **azione integrale** per disturbi costanti a regime, abbiamo semplicemente usato il comando ***lqi()*** ridefinendo le matrici di peso per trovare il **nuovo guadagno** per la parte **LQR**:

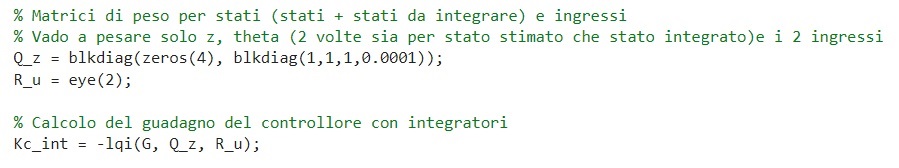


Figura 2.5 – Guadagno per controllo LQG integrale (con azione integrale su e )

**3.1 Controllori** **Mixed-Sensitivity e H-inf**

Il primo passo effettuato per andare a ricavare il controllore **Mixed-Sensitivity e H-inf** è stato **linearizzare** il sistema intorno al **punto di equilibrio** su cui vogliamo andare a **controllarlo** (come già visto precedentemente nell’analisi di Raggiungibilità e Osservabilità).

Quello di seguito è lo schema a blocchi usato per trovare il controllore Mixed-Sensitivity e quello H-inf:

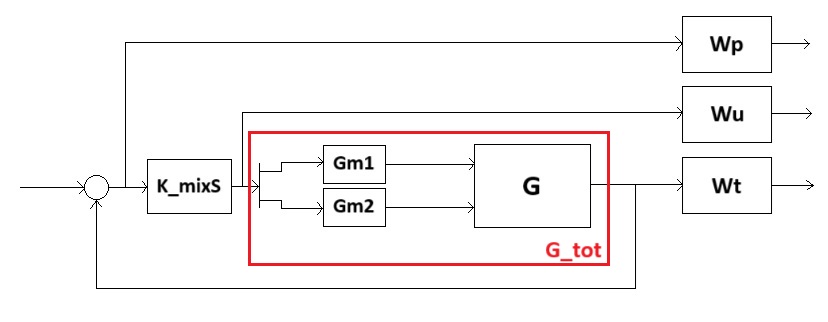
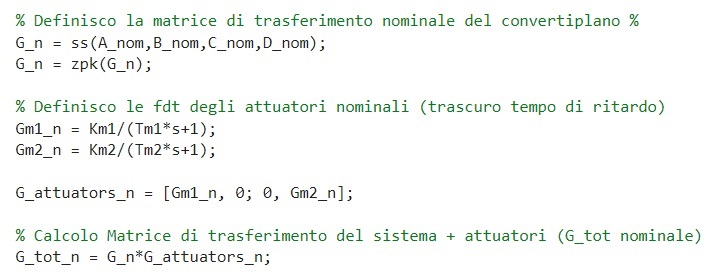
****

Figura 3.1 – Schema a blocchi per Mixed-Sensitivity

Per trovare il sistema totale (la matrice P nella forma di Doyle senza i pesi), che vediamo in Figura 3.1 come il sistema G\_tot, abbiamo eseguito il seguente codice:

Figura 3.2 – Codice per calcolo G\_tot nominale



Il prossimo passo per la sintesi di un controllo è quello di definire i pesi di **Prestazione**(WP), **Robustezza**(WT) e **sforzo di controllo**(WU): abbiamo scelto come peso WU una costante, invece per la scelta di WP ci siamo basati sulla Sensitività del controllo LQG; di conseguenza la scelta di WT è automatica, cioè complementare a WP.

Quindi abbiamo scelto WP utilizzando il controllore LQG già sintetizzato:

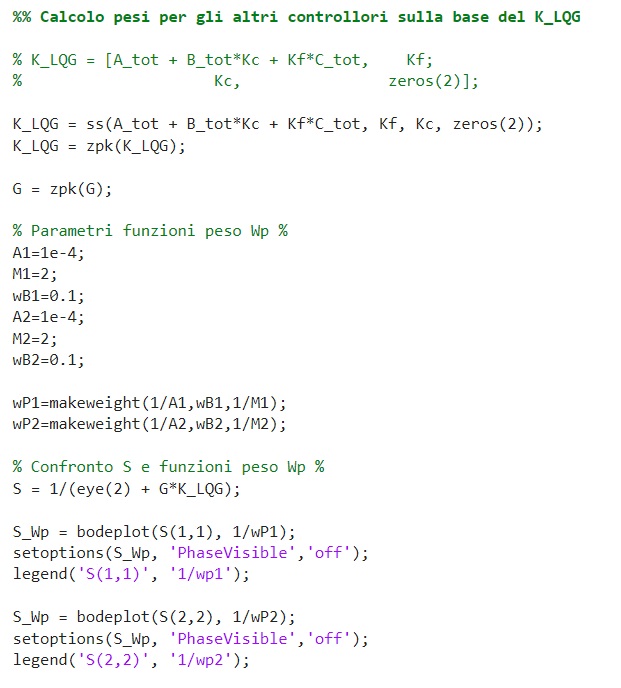


Figura 3.3 – Codice per ricavare il peso WP

Che con i parametri per i pesi WP scelti in *Figura 3.3*, troviamo:

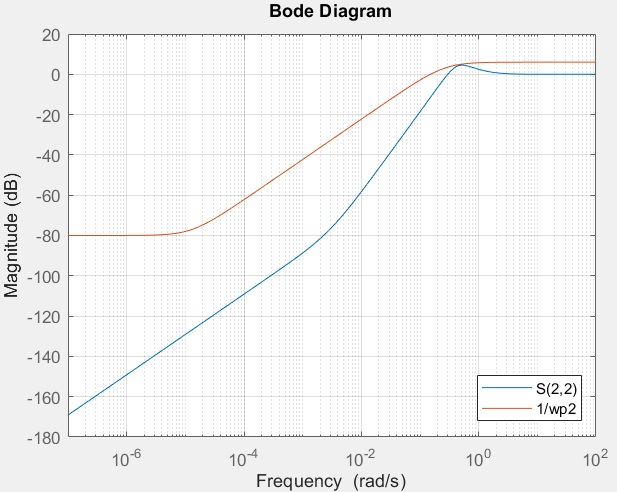
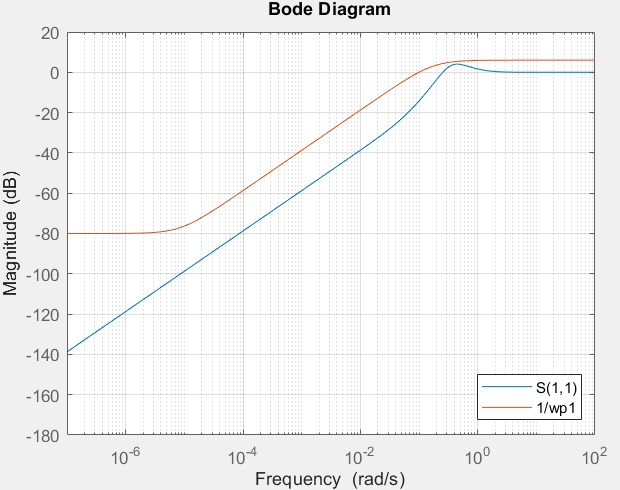
****

Figura 3.4 – Scelta del peso WP

A questo punto, definiti tutti i pesi, possiamo sintetizzare il controllore con il comando ***mixsyn()***:

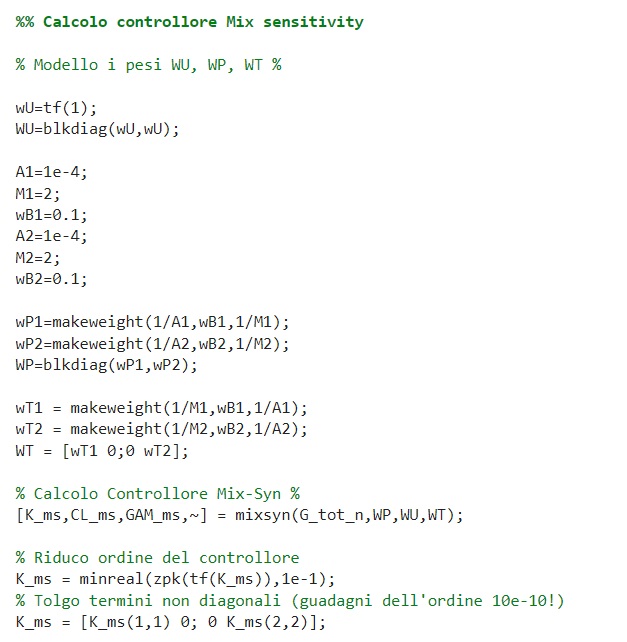
****

Figura 3.5 – Calcolo Pesi e calcolo controllore approssimato

Invece per sintetizzare un controllore H-inf bisogna usare il comando ***hinfsyn()***, per cui servono i pesi usati nel Mixed-Sensitivity (WP, WT e WU) e mettere il sistema nella forma P-K usando il comando ***connect()***. Infatti, abbiamo definito ingressi e uscite per ciascun blocco nel seguente modo:

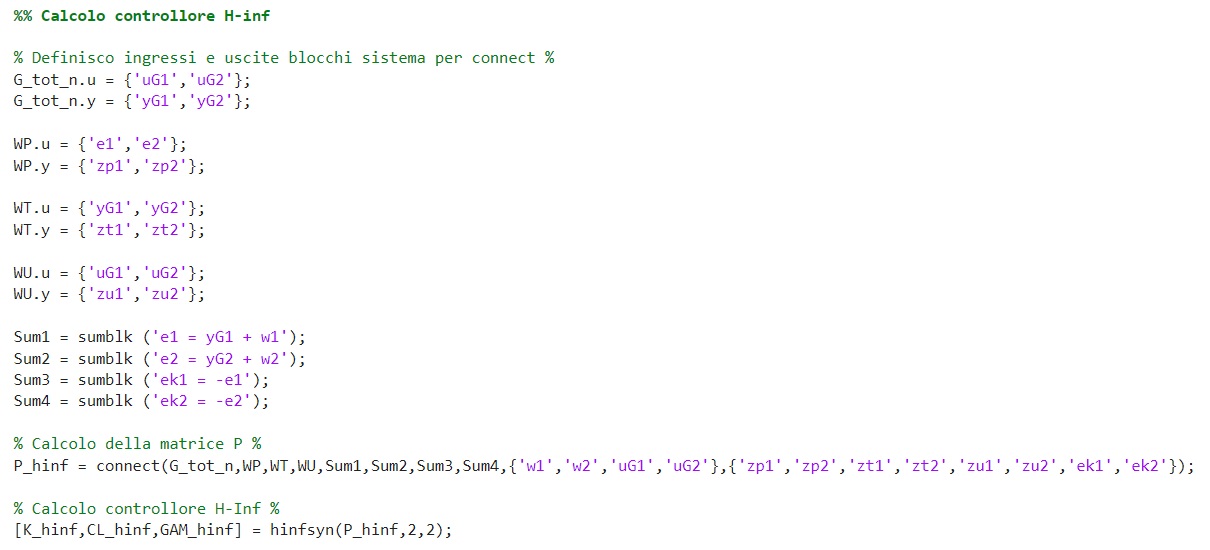


Figura 3.6 – Configurazione ingressi e uscite e calcolo della matrice P

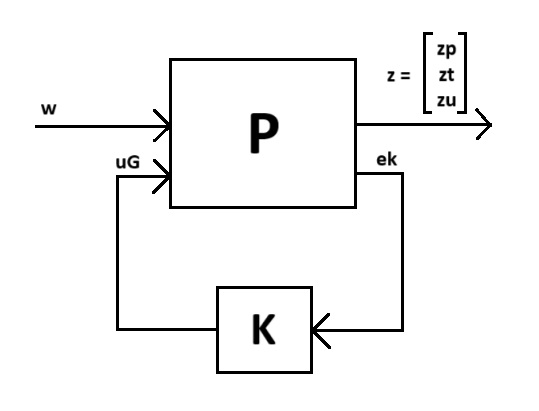
****

Figura 3.7 – Schema a blocchi per sintesi controllore H-inf

Infine, abbiamo calcolato il controllore con il comando sopra menzionato:

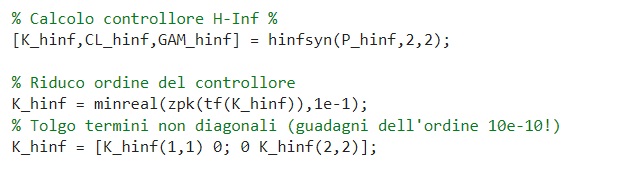


Figura 3.8 – Codice sintesi controllore H-inf approssimato

**4.1 Controllore** ***µ*-Synthesis con DK-iteration**

Il primo passo effettuato per andare a ricavare il controllore ***µ*-Synthesis** è stato **linearizzare** il sistema intorno al **punto di equilibrio** su cui vogliamo andare a **controllarlo** (come già visto precedentemente nell’analisi di Raggiungibilità e Osservabilità).

Linearizzato il sistema in forma di stato, lo scriviamo come matrice di trasferimento G e relativa matrice di incertezza (avendo definito come parametri incerti e di ):



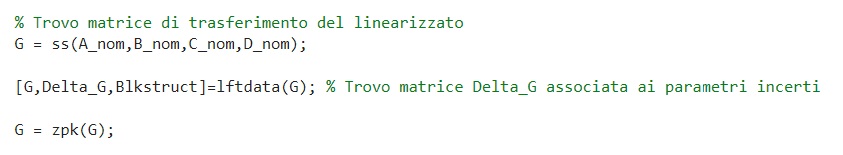


Figura 4.1 – Uso del comando **lftdata()** per trovare e G

Ora serve calcolare i pesi di incertezza degli attuatori e , utilizzando l’incertezza moltiplicativa (concentrata), dovuta dal fatto che nel modello nominale abbiamo trascurato il ritardo:

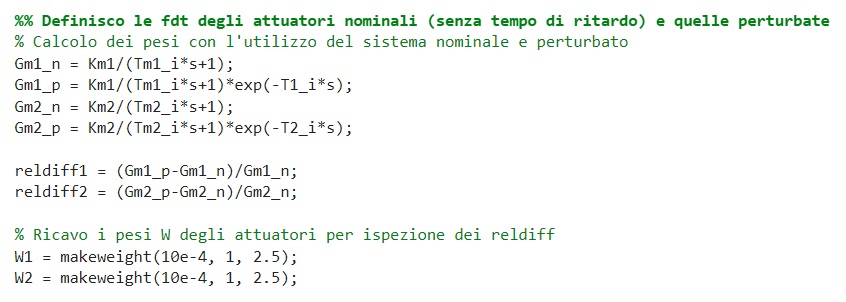


Figura 4.2 – Calcolo dei pesi e per ispezione

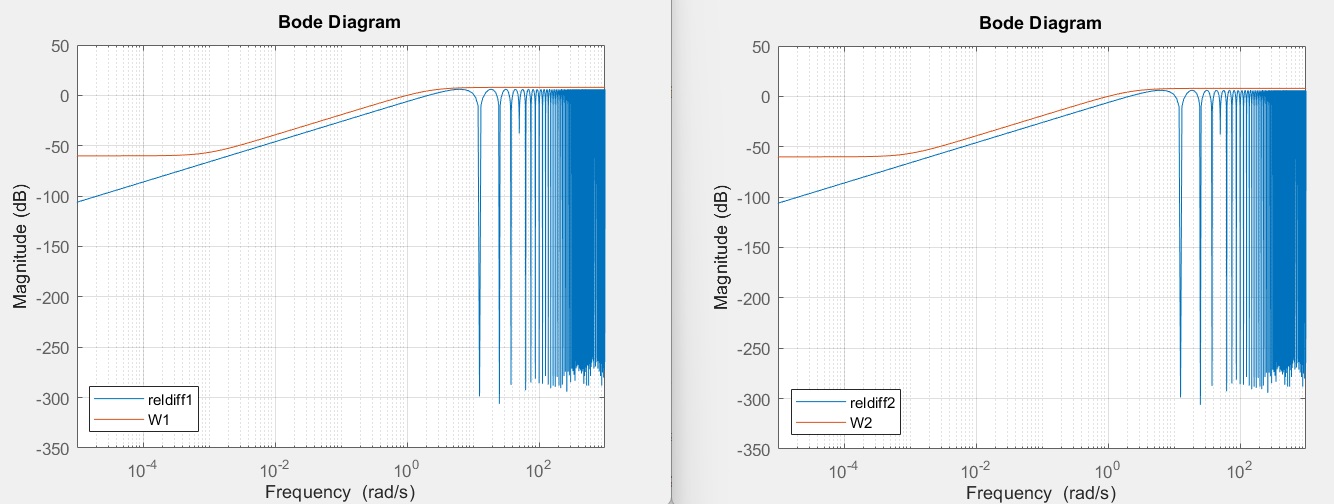


Figura 4.3 – Pesi e

A questo punto siamo arrivati ad avere il seguente schema a blocchi (considerando che il WP è uguale a quello per la Mixed-Sensitivity):

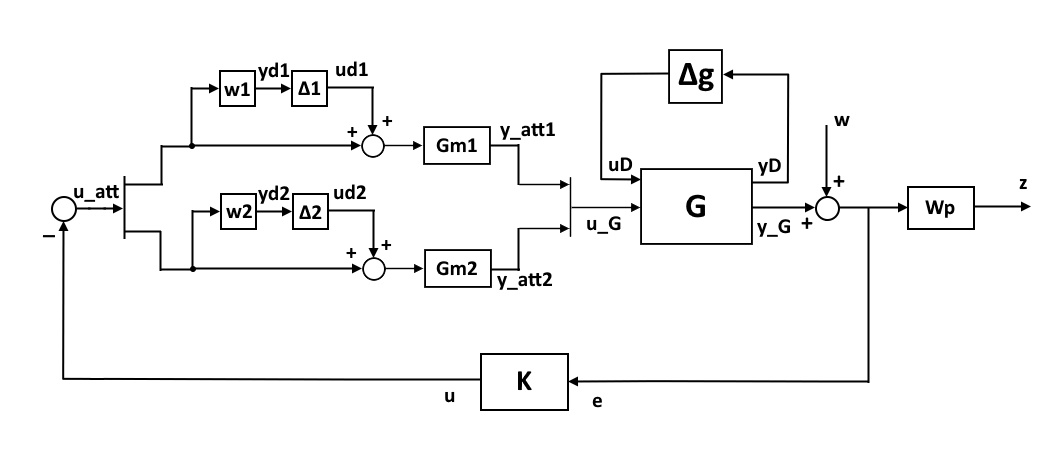
****

Figura 4.4 – Schema a blocchi per la sintesi del controllo con µ-Synthesis

Ora cerco di tirare fuori un blocco di incertezza strutturato degli attuatori, così da avere uno schema più semplice da gestire con cui usare il comando ***connect()***; lo schema risultante e il codice che lo realizza sono i seguenti:

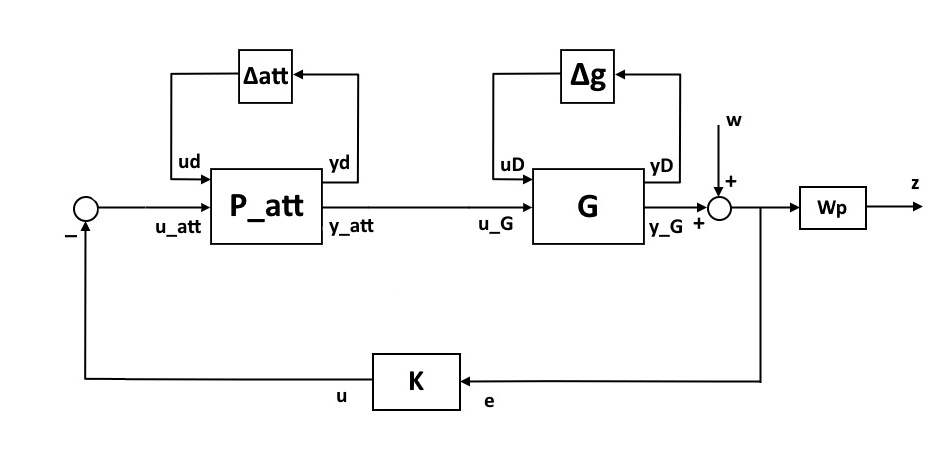


Figura 4.5 – Schema a blocchi dopo aver tirato fuori

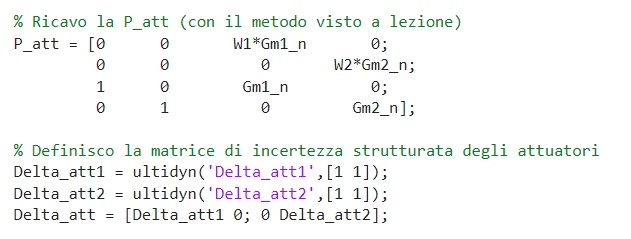


Figura 4.6 – Codice per ricavare e

In questo modo, ora possiamo mettere tutto insieme per trovare la forma di Doyle Δ-P-K, dove dentro la matrice ci sono , e ; invece la matrice è fatta come segue:

Il codice che realizza ciò che abbiamo appena spiegato:



Figura 4.7 – Codice per ricavare P e costruzione di

Da cui segue la sintesi del controllore con il comando ***musyn()***:

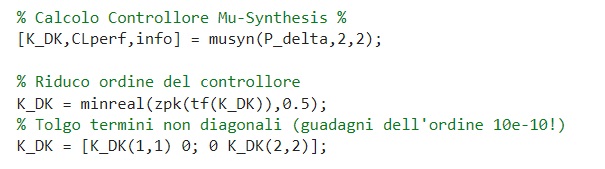


Figura 4.8 – Codice per sintesi del controllore approssimato

**5.1 Risultati LQG**

Abbiamo fatto le seguenti prove con il controllore LQG:

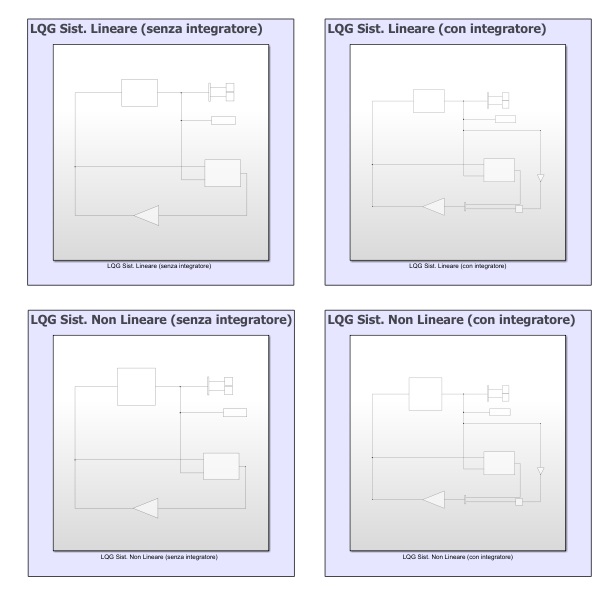


Figura 5.1 – Prove LQG

Abbiamo fatto diverse prove con le varie configurazione mostrate qui sopra, andando a cambiare le condizioni iniziali così da trovare l’esistenza di una **RAS** che fa convergere il sistema nel punto di equilibrio:

* **Prova 1**: condizioni iniziali nulle (punto di equilibrio)
* **Prova 2**: , , ,
* **Prova 3**: , , ,

Il risultato finale è che tutte le configurazioni hanno l’equilibrio scelto **Globalmente Asintoticamente Stabile**, a meno del sistema Non Lineare senza integratore che è solo Marginalmente stabile per lo stato :

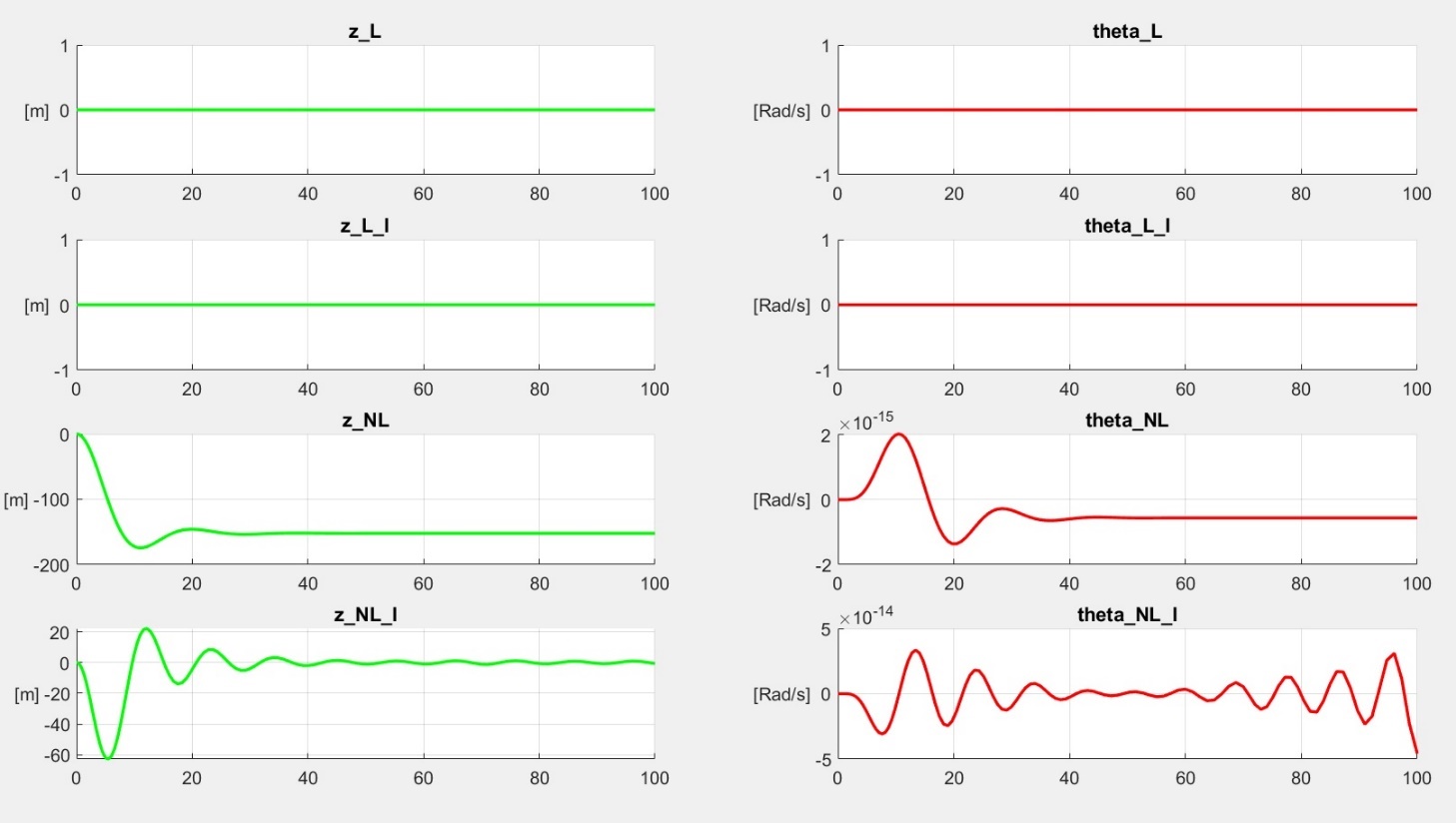


Figura 5.2 – **Prova 1**

Si può constatare infatti che la non converge a zero.

Questo risultato ce lo aspettiamo perché nel caso **Non Lineare Senza Integratore** abbiamo che non riusciamo a reiettare la costante nella prima equazione dinamica che caratterizza l’evoluzione dello stato : non avendo un integratore, l’errore a regime non potrà convergere quindi a zero.

Abbiamo aumentato il tempo di simulazione fino a per capire se l’andamento di nel caso Non Lineare divergesse:

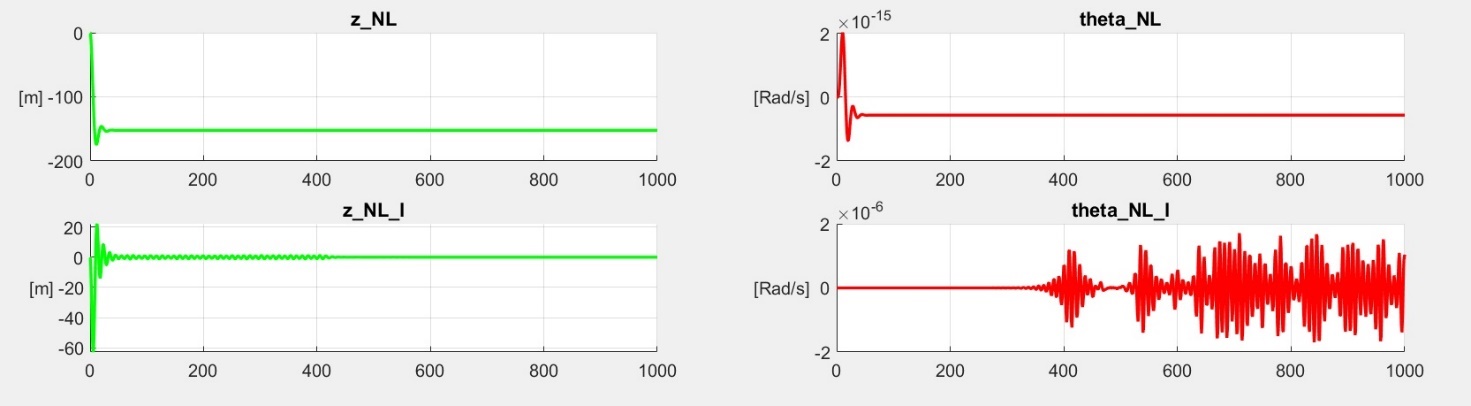


Figura 5.3 – **Prova 1** (osservazione dello stato nel caso Non Lineare con Integratore)

Ora vediamo le altre prove:

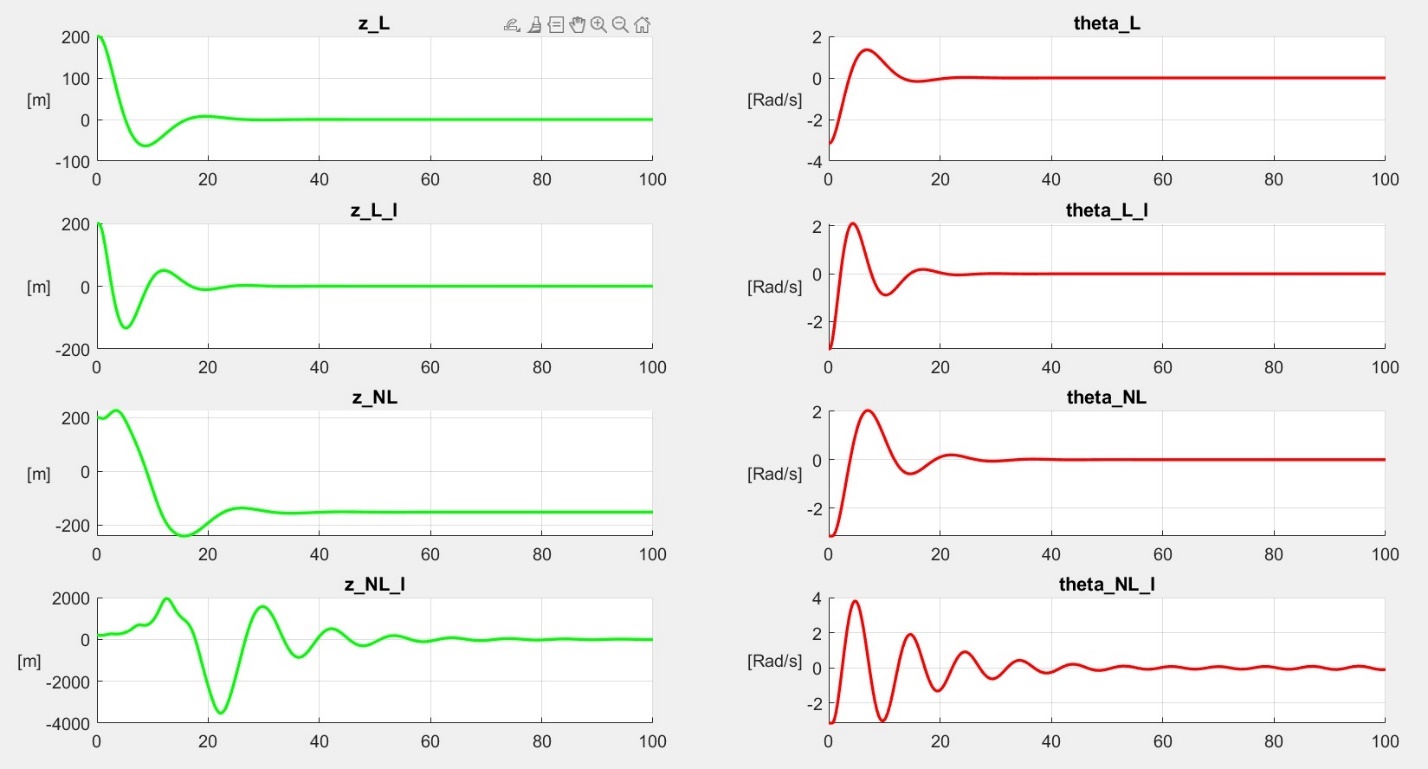


Figura 5.4 – **Prova 2** (; )

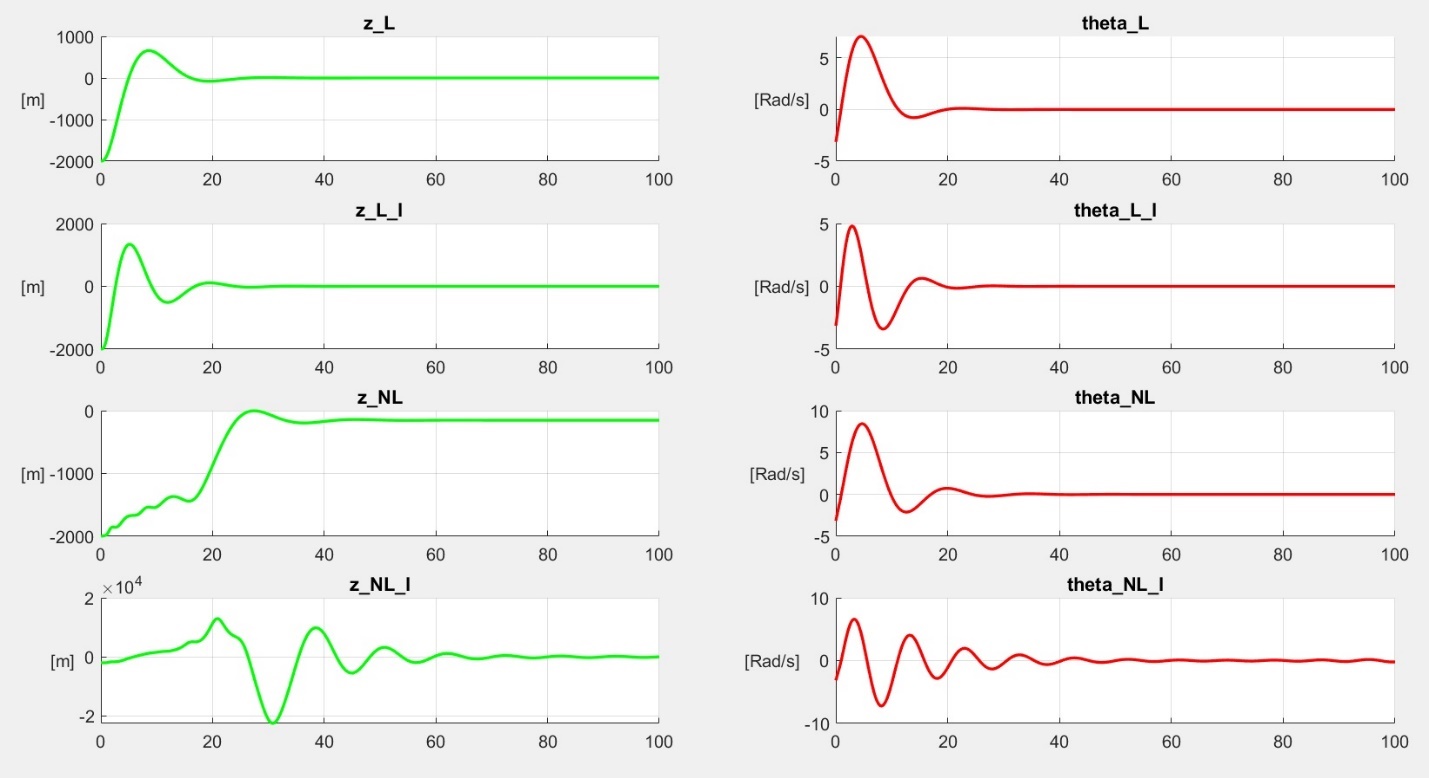


Figura 5.5 – **Prova 3** (; )

Quindi, dato che nel caso **Non Lineare Senza Integratore** abbiamo che comunque lo stato converge asintoticamente a zero, non serve applicare un integratore per quello stato, anche perché va solo a peggiorare le performance (vedi *Figura 5.4* e *Figura 5.5*); perciò abbiamo ridotto quasi a zero il peso sull’azione integrale sul singolo stato così da avere:

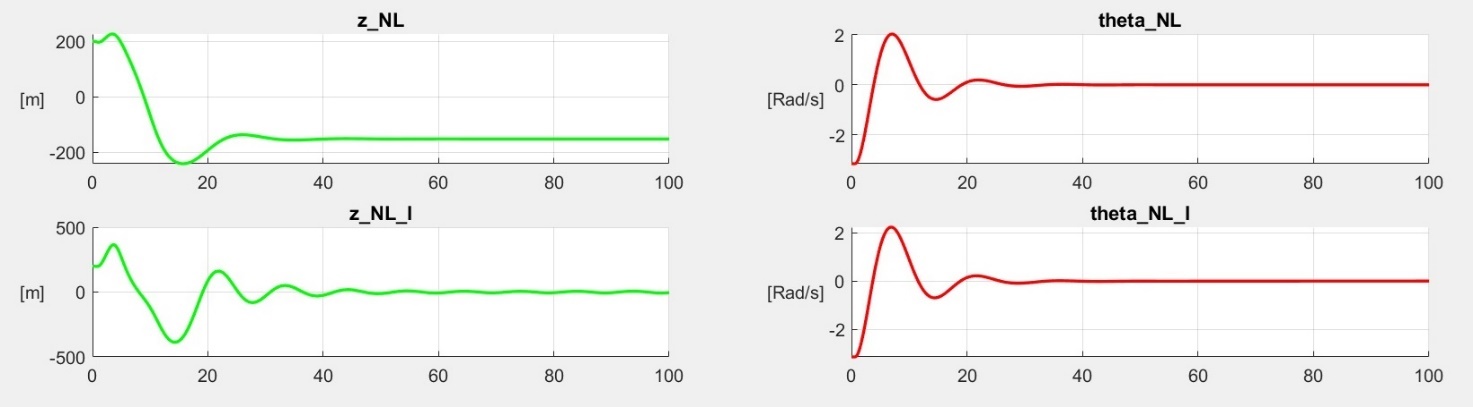


Figura 5.6 – **Prova 2** togliendo azione integrale a

E’ da notare che anche le performance dello stato sono migliorate.

Abbiamo poi cercato di migliorare la stabilità (reiezione overshoot) del controllore LQG cercando di andare a pesare maggiormente (scelto un fattore 100) gli ingressi di controllo e rispetto agli stati e (mi aspetto di convergere al punto di equilibrio più lentamente):

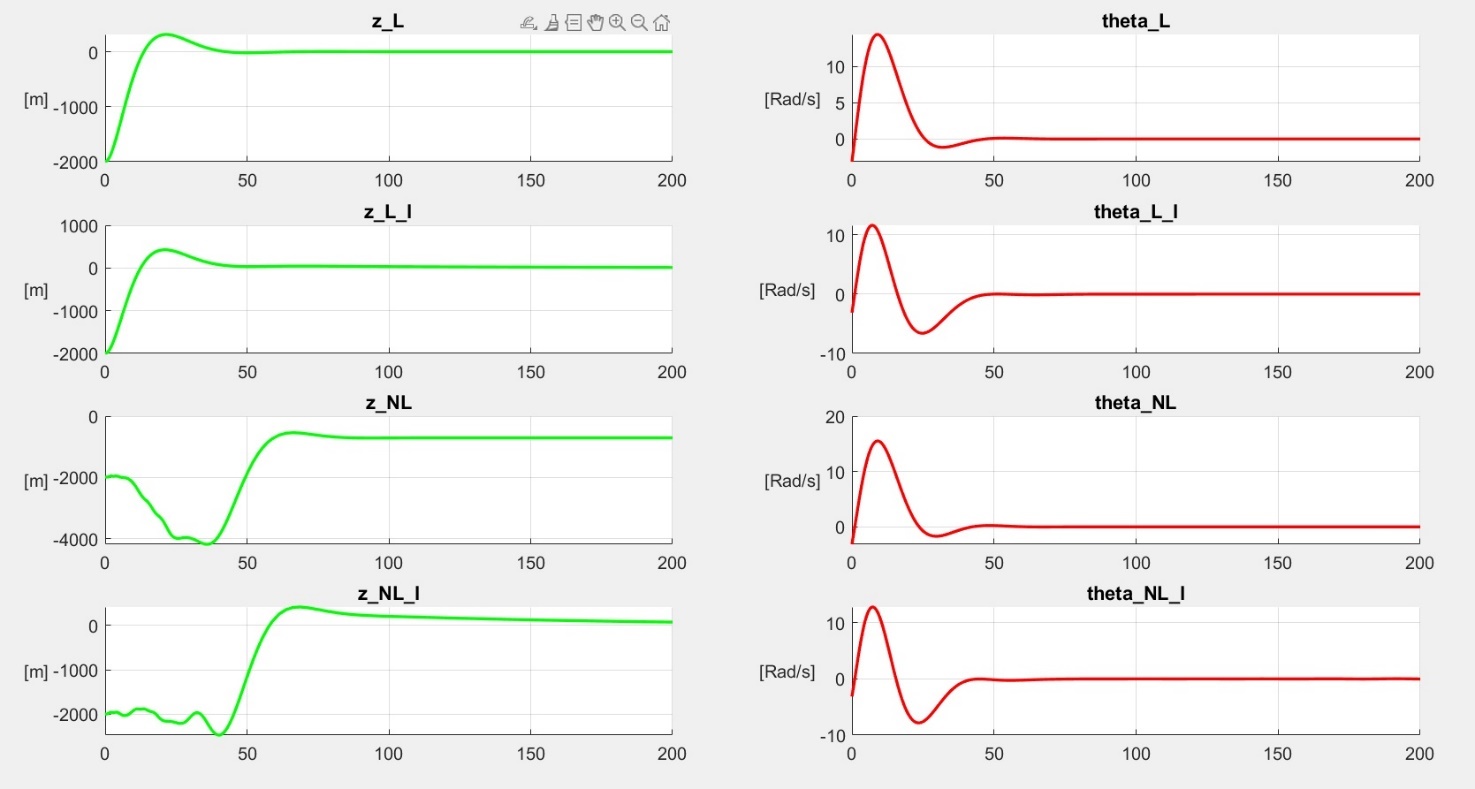


Figura 5.7 – **Prova 3** abbassando azione integrale a e (; )

Possiamo quindi vedere un andamento più smooth per tutte e 4 le configurazioni: abbiamo guadagnato in stabilità di convergenza, ma abbiamo perso nelle prestazioni di tracking.

Infatti, se prima avevamo un **tempo di assestamento** di , ora invece è aumentato in un intervallo di .

**6.1 Risultati Mixed-Sensitivity e H-inf**

**7.1 Risultati *µ*-Synthesis con DK-iteration**