POLITECNICO DI MILANO

Scuola di Ingegneria Industriale e dell'Informazione

Corso di Laurea Magistrale in
Ingegneria dell'Automazione

Controllo Predittivo non Lineare di un Manipolatore Aereo

Relatore: Prof. Luca BASCETTA

Correlatore: Dott. Roberto ROSSI

Tesi di Laurea di:

Dario LUNNI Matr. 824360

Anno Accademico 2014 - 2015

Politecnico di Milano:
www.polimi.it

Scuola di Ingegneria Industriale e dell'Informazione:
www.ingindinf.polimi.it

Indice

Introduzione **1**

1 Introduzione **1**
1.1 Stato dell'arte 4
1.2 Struttura della Tesi 9

2 Appartato Sperimentale e Modello Matematico **11**
2.1 Drone 11
2.2 Manipolatore 13
2.3 Microcontrollore 14
2.4 Architettura di comunicazione 18
2.5 Cinematica 20
2.6 Dinamica 26
2.6.1 Drone 26
2.6.2 Braccio robotico 29
2.6.3 Sistema complessivo 29

3 Architettura di Controllo **33**
3.1 Introduzione 33
3.2 Architettura di controllo 33
3.3 Model Predictive Control 35
3.3.1 Soluzione in anello aperto 36
3.3.2 Receding Horizon 39
3.3.3 Inseguimento di riferimenti 40
3.3.4 Stabilità 40
3.3.5 Formulazione non lineare 43

3.4 Applicazione del Controllo Predittivo al sistema 45
3.4.1 Modello interamente dinamico 45
3.4.2 Modello ibrido 46
3.4.3 Modello interamente cinematico 47
3.4.4 Funzionale di costo 48
3.4.5 Vincoli 48
3.4.6 Introduzione della gerarchia 51
3.5 Controllo di basso livello 53
3.5.1 Controllo di assetto 53
3.5.2 Controllo di posizione 54
3.5.3 Controllo del braccio robotico 56

4 Simulazioni **57**
4.1 Introduzione 57
4.2 Simulazioni in ambiente Matlab 57
4.2.1 Modello Completamente Dinamico 58
4.2.2 Modello Ibrido 58
4.2.3 Modello Cinematico 62
4.2.4 Modello Cinematico con Gerarchia 64
4.3 Simulazioni Software in The Loop 67
4.3.1 ROS - Robot Operating System 67
4.3.2 Simulatore Gazebo 67
4.3.3 Simulazioni in ambiente Gazebo 69

5 Prove Sperimentali **75**
5.1 Introduzione 75
5.2 Relazione tra variabili di controllo e forze aerodinamiche . 75
5.3 Controllo di assetto del Drone 80
5.4 Controllo di posizione del Drone 85
5.5 Controllo MPC applicato al solo braccio robotico 87

Conclusioni **93**
5.6 Sviluppi futuri 94

A Taratura anelli di controllo di basso livello **95**
A.1 Anello di controllo di assetto e posizione del drone 95
A.2 Anello di controllo del braccio robotico 97

Bibliografia **99**

Elenco delle figure

1.1 Drone per la consegna di pacchi. 2
1.2 Drone utilizzato nel progetto ARCAS. 5
1.3 Afferraggio *avian-inspired*. 7
1.4 Applicazioni di costruzione. 8

2.1 Ottocottero X8+. 12
2.2 CrustCrawler. 14
2.3 Pixhawk. 15
2.4 Odroid XU3 16
2.5 Schema di comunicazione tra i diversi componenti 18
2.6 Comunicazione tra le componenti 19
2.7 Disposizione delle terne di riferimento. 20
2.8 Disposizione delle terne del braccio robotico. 22
2.9 Posizione dei Motori 28

3.1 Architettura di controllo complessiva. 34
3.2 Schema di controllo NMPC 44
3.3 Rappresentazione del vincolo per evitare la collisione. . . . 49
3.4 Interazione dei sistemi di controllo di alto e basso livello per il controllo del drone. 53
3.5 Anelli di Attitude Control. Dopo i blocchi dei regolatori troviamo il blocco rappresentante il sistema (nel caso di un solo grado di libertà contenente il solo effetto dell'inerizia.) 54

3.6 Anelli di controllo di posizione. Dopo i blocchi dei regolatori troviamo il blocco H rappresentante il controllo di assetto (approssimato come un filtro passa basso) che ci permette di seguire il riferimento di forza che utilizziamo come variabile di controllo nell'anello di posizione. 55
3.7 Sistema di controllo di posizionamento del braccio. 56

4.1 Rappresentazione 3D della traiettoria inseguita. 59
4.2 Rappresentazione della traiettoria inseguita dall'end effector. 60
4.3 Tempi computazionali di simulazione. Durante ciascuno step vengono calcolati i valori ottimi delle variabili di controllo sui 10 step futuri. 61
4.4 Traiettoria dell'*end-effector*. 63
4.5 Errore di posizionamento dell'*end-effector*. 63
4.6 Traiettoria dell'*end-effector*. 64
4.7 Errore di posizionamento dell'*end-effector*. 65
4.8 Disallineamento dei baricentri con e senza approccio gerarchico. 66
4.9 Tempo computazionale di ottimizzazione. 66
4.10 Modello 3D di drone Iris. 69
4.11 Architettura ROS in simulazione. 70
4.12 Architettura ROS in simulazione con generazione di traiettoria. 71
4.13 Risposta al gradino in presenza o assenza di generazione di traiettoria. 73
4.14 Risposta del sistema ad un riferimento costante sul piano xy 74
4.15 Risposta del sistema ad un riferimento costante in z 74

5.1 Banco di prova per testare il controllo di assetto. 80
5.2 Risposta allo step dell'anello di controllo di rollio. 81
5.3 Ingrandimento della risposta al gradino dell'anelli di controllo di rollio. 81
5.4 Setup sperimentale per test di volo. 82
5.5 Condizione del drone durante la prova. 83
5.6 Risposta al gradino dell'anello di controllo d'imbardata. . . 84

5.7 Risposta al gradino dell'anello di controllo d'imbardata. . . 84
5.8 Risposta al gradino dell'anello di controllo di altitudine . . 86
5.9 Risposta al gradino dell'anello di controllo di posizione. . . 87
5.10 Andamento della risposta dell'*end-effector* nelle coordinate xyz. 90
5.11 Riferimenti generati e risposta delle variabili di giunto. . . 90
5.12 Vincoli sull'attuazioni. 91
5.13 Posizione dell'*end-effector* al termine della prova. 92
5.14 Posizione dell'*end-effector* al termine della prova. 92

A.1 Diagramma di Bode: anelli per il controllo della posizione angolare. 96
A.2 Diagramma di Bode: anelli per il controllo della posizione xy. 97
A.3 Diagrammi di Bode: anelli per il controllo della posizione dei giunti. 98

Elenco delle tabelle

2.1 Specifiche tecniche dei servocomandi Dynamixel 15
2.2 Parametri Denavit-Hartemberg per il manipolatore robotico a 4 GdL 21

4.1 Parametri dinamici utilizzati in ambiente Matlab. 58

Sommario

La presente tesi si occupa dello studio del problema di controllo di un manipolatore aereo autonomo (UAM), ovvero un drone equipaggiato con un manipolatore robotico. Lo studio ha condotto allo sviluppo di un Controllo Predittivo non Lineare con l'obiettivo di inseguire una traiettoria per l'end-effector. Sono stati sviluppati tre modelli di predizione per il sistema, di cui sono state analizzate le performance di controllo e computazionali. Il primo modello considera la dinamica completa del sistema, mentre il secondo modello tiene conto della dinamica del solo drone e utilizza un modello cinematico per il braccio robotico, infine il terzo utilizza un modello cinematico di tutto il sistema, e permette di ottenere più bassi tempi computazionali. Nel contesto del Controllo Predittivo cinematico è stata inoltre implementata, in modo gerarchico rispetto all'obiettivo primario, la minimizzazione del disallineamento dei baricentri, di drone e braccio robotico. I risultati dello studio sono stati simulati in ambiente Matlab, validando l'architettura di controllo costruita. Ciascuna tipologia di Controllo Predittivo viene accoppiato con un algoritmo di controllo di basso livello che permette l'inseguimento dei riferimenti generati sia per il drone che per il braccio robotico. Sono stati inoltre studiati, tarati e validati i controlli di basso livello necessari per implementare le diverse tipologie di Controllo Predittivo. Si è implementato il Controllo Predittivo cinematico per solo drone sul sistema operativo ROS, validandolo tramite simulazioni Software in The Loop. Si sono infine condotte prove sperimentali per la stima di alcuni parametri del sistema reale, la validazione dei controlli di basso livello e di MPC per il solo braccio.

Parole chiave: Controllo Predittivo, Manipolazione Aerea, Quadricottero.

Abstract

This thesis addresses the problem of controlling an unmanned aerial manipulator (UAM), e.g. a quadrotor with a serial arm endowed below. To that end we developed a Non linear Model Predictive Control (NMPC) whose main target consists in the following of a trajectory using the end-effector arm. Three different UAM models are developed. The first and more complex model considers both the quadrotor and arm dynamic. In a second model, we consider the the arm kinematically, but still model dynamically the quadrotor. The third model considers only the quadrotor and arm kinematics, allowing for fast computation. We develop a hierarchical approach for the problem too. In this hierarchy, a main task consists on following a desired trajectory with the arm end-effector while a secondary task vertically aligns the arm CoG to that of the vehicle during the flight. The hierarchical approach here proposed is implemented through a cascade of constrained optimizations. Results of the study are simulated using Matlab, and validate the control architecture. Every model is coupled with a tuned low level controller that allows the system to follow the generated references. The MPC that use the kinematic for the quadrotor only has been implemented using ROS and some Software in the Loop Simulations have been carried out. Finally, experimental tests have been conducted for the estimation of certain parameters of the real system and the validation of part of the control.

Keywords: Model Predictive Control, Aerial Manipulation, Quadcopter.

Capitolo 1

Introduzione

Negli ultimi anni si sta diffondendo sempre più l'utilizzo di piccoli veicoli aerei senza pilota. Si tratta essenzialmente di robot volanti che prendono il nome di UAV (Unmanned Aerial Vechicle), o droni, che vengono utilizzati per eseguire diversi compiti. Tra le applicazioni più comuni si trovano: operazioni di sorveglianza e ispezione di luoghi difficilmente accessibili, misurazioni di diverso tipo (ad esempio qualità dell'aria), ed in particolare in situazioni di emergenza da luoghi non agevolmente raggiungibili. Sono state realizzate diverse tipologie di droni in grado di assolvere questi compiti, le quali si possono dividere in diverse categorie. Innanzitutto in base al principio di funzionamento si differenziano droni ad ala fissa, simili a piccoli aerei, e droni ad ala rotante, simili a piccoli elicotteri e dotati di due o più rotori. In secondo luogo, nel campo dei droni ad ala rotante si possono individuare diversi gruppi in funzione del numero di rotori di cui i robot sono dotati. In particolare una delle tipologie di droni ad ala rotante più diffuse ed utilizzate è quella dei quadricotteri, grazie all' elevata destrezza e al basso costo che li caratterizza. Il controllo di questi droni è stato studiato approfonditamente negli ultimi dieci anni, ma è ancora un ambito di ricerca di attualità a causa del continuo incremento delle prestazioni acrobatiche che possono essere raggiunte.

Recentemente la comunità di ricerca sui droni ha spostato il proprio interesse da applicazioni "passive", esposte precedentemente, ad applicazini

"attive" di interazione con l'ambiente. Per conseguire questi obiettivi si equipaggia la base del drone con un manipolatore robotico, componendo in questo modo un manipolatore aereo, o UAM (Unmanned Aerial Vehicle). L'utilizzo di un braccio robotico vincolato ad una base mobile permette di ampliare l'area di applicazione dei droni. Alcune applicazioni possibili sono ad esempio:

- operazioni di interazione in ambienti pericolosi o difficili da raggiungere per l'uomo,
- operazioni di asservimento visivo,
- raccolta e trasporto di campioni di misura,
- trasporto di medicinali in situazioni di emergenza,
- costruzione di strutture.

Alcune di queste tecnlogie sono già in sperimentazione in alcune aziende, poichè in grado di semplificare e rendere più economiche operazioni normalmente affidate ad operatori specializzati. Ad esempio si può citare l'utilizzo dei droni per la consegna di pacchi da parte di grandi aziende di e-commerce e trasporti (Amazon o DHL), o l'utilizzo dei manipolatori aerei in occasione di eventi catastrofici.

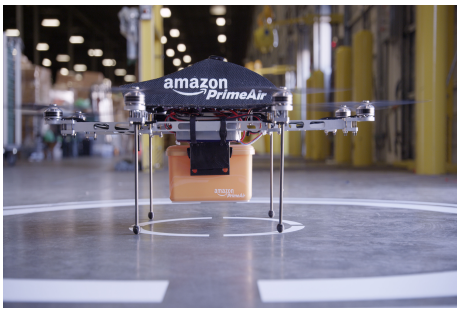


Figura 1.1: Drone per la consegna di pacchi.

Essendo un ambito di attualità e soggetto a forte crescita, il controllo di un manipolatore aereo è quindi di grande interesse per la comunità scientifica. L'obiettivo della presente tesi, in particolare, è la sintesi di

un algoritmo di controllo che permetta, data una traiettoria desiderata dell'*end-effector* del manipolatore aereo, di calcolare le variabili di riferimento per le variabili di stato. Data la complessità del sistema, lo studio della strategia di controllo ha portato alla sintesi di un controllore di alto livello. Nello specifico, l'oggetto della tesi è l'applicazione di una tecnica di controllo predittivo non lineare al sistema complessivo. Nell'ambito della robotica l'applicazione di questo algoritmo è di grande interesse negli ultimi anni, poichè lo sviluppo delle tecnologie elettroniche ha da poco permesso il raggiungimento di prestazioni sufficienti da permettere l'applicazione di questo algoritmo ai dispositivi meccatronici.

Lo studio dell'algoritmo è stato affrontato da un punto di vista teorico e poi implementato in simulazione. I risultati delle simulazioni portano alla validazione dell'analisi e della sintesi del controllo sotto studio. Infine alcune prove sperimentali sono state effettuate allo scopo di verificare il funzionamento di parte del sistema di controllo realizzato.

Il progetto è stato condotto in collaborazione con l'istituto di ricerca IRI (*Istitut Robòtica i Informàtica Industrial*), un istituto di ricerca della *Universitat Politècnica de Catalunya* (UPC). In particolare, la parte di studio teorico e di implementazione in simulazione è stata condotta nell'istituto stesso.

1.1 Stato dell'arte

Il problema di controllo di un manipolatore aereo rientra all'interno dell'insieme più ampio del problema del controllo dei manipolatori robotici a base mobile. In questo ambito l'aspetto del controllo è da sempre di difficile soluzione per il progettista. Innanzituto perchè il sistema, per il corretto funzionamento, deve essere sottoposto a numerosi vincoli (accelerazione, velocità, posizione dei giunti ecc.), in secondo luogo perchè il robot si trova immerso in un ambiente dinamico con il quale interagisce. Nel contesto particolare della manipolazione aerea, essendo la base mobile costituita da un drone, deve essere garantita la stabilizzazione del velivolo anche durante il movimento del braccio robotico. Uno dei principali problemi, a cui si va incontro nella gestione di un manipolatore aereo, è dunque il controllo del sistema drone. Questo è infatti sottoposto ai disturbi dati dal movimento del braccio robotico a esso connesso.

Negli ultimi anni sono stati sviluppati diversi approcci per l'efficiente controllo dei quadricotteri. Gli approcci classici, come le diverse varianti di PID [1], [2], [3], LQR [4], [5] o controllo robusto H_∞ [6] sono basati sulla linearizzazione del sistema nell'intorno di una posizione di equilibrio. Ciò permette un buon controllo del robot in prossimità di tale posizione, ma le prestazioni decadono se il sistema si allontana da questa condizione. Per superare queste limitazioni, e ottenere migliori performace, sono state studiate numerose tecniche di controllo non lineare. Si citano come esempio: tecniche di *backstepping* [7], controllo adattivo [8], *sliding mode* [9]. Altri studi presenti in letteratura fanno uso di tecniche di controllo predittivo a breve orizzonte di predizione per ridurre la complessità computazionale del problema [10].

Per consentire l'assemblaggio di un manipolatore aereo è stato fondamentale studiare l'effetto del movimento del braccio robotico sullla stabilità del sistema complessivo [11], [12]. I primi lavori legati alla manipolazione aerea hanno riguardato lo studio della dinamica durante il trasporto di carchi tramite elicotteri [13]. Si è passati poi allo studio delle operazioni di

Figura 1.2: Drone utilizzato nel progetto ARCAS.

afferraggio da parte del drone tramite *end-effector*, e agli effetti di queste operazioni sulla dinamica sia in condizione di *hovering* che in condizione di movimento. Più recentemente si è spostata l'attenzione sulla costruzione in autonomia di strutture e sull'interazione con l'ambiente. Il progetto Aerial Robotics Cooperative Assembly System (ARCAS) [14] a altri progetti [15] hanno dimostrato il raggiungimento di complessi obiettivi di manipolazione usando manipolatori a elevato numero di gradi di libertà. Altre importanti obiettivi raggiunti sono stati lo sviluppo di tecniche di afferraggio *avian-inspired* [16], nelle quali le operazioni di afferraggio avvengono con velocità del drone di 2 m/s. Riguardo alle tecniche di controllo sono stati sviluppati diversi approcci. Si citano: tecniche di controllo gerarchico che sfruttano l'inversione cinematica per generare riferimenti da inviare ai controllori di basso livello [17], oppure controlli di impedenza per permettere l'efficiente interazione con l'ambiente [18], o ancora tecniche di *feedback linearization* per l'inseguimento di traiettorie [19].

Per i droni equipaggiati con un braccio robotico, e in generale per i manipolatori a base mobile, è fondamentale poter generare una traiettoria per le variabili di stato del sistema che non interferisca con i vincoli necessari al corretto funzionamento del sistema stesso. Solitamente ciò che accade nell'ambito dei manipolatori a base mobile è l'utilizzo di controlli di alto livello basati su ottimizzazioni vincolate risolvibili online [20], [21], con l'obiettivo di risolvere il problema del controllo dal punto di vista cine-

matico. Principalmente le possibili soluzioni al problema possono essere: considerare l'obiettivo che deve essere raggiunto come un vincolo che il robot deve sempre soddisfare mentre viene minimizzata una funzione di costo [22], oppure formulare il l'obiettivo stesso come funzionale sottoposto a minimizzazione. Durante la generazione della traiettoria possono essere minimizzati funzionali legati ad esempio alla manipolabilità del braccio robotico, oppure possono essere introdotti vincoli per evitare eventuali ostacoli [23], [24].

Nell'ambito dei veicoli autonomi la generazione della traiettoria è un ambito di ricerca molto attuale, soprattutto per quanto riguarda la realizzabilità di una traiettoria. In particolare, in letteratura sono presenti alcuni esempi di applicazione degli algoritmi di pianificazione di traiettoria in *real-time* nell'ambito dei quadricotteri. Esistono algoritmi che generano una traiettoria nel dominio dello spazio a partire da un insieme di funzioni primitive, ed in seguito ottimizzano la velocità di esecuzione in modo che i vincoli imposti al sistema non vengano violati utilizzando tecniche di ottimizzazione non lineare [25]. Altri algoritmi di generazione di traiettoria *real-time* hanno come obiettivo la minimizzazione dello snap attraverso una sequenza di posizioni, e garantiscono allo stesso tempo il soddisfacimento di vincoli su stato e attuatori [26]. Altri algoritmi ancora hanno l'obiettivo di generare traiettorie che minimizzino il tempo di raggiungimento, ma allo stesso tempo siano realizzabili [27].

Al contrario del campo dei quadricotteri, nel campo dellla manipolazione aerea, la generazione di traiettoria non è ancora stata studiata in profondità. In particolare si trovano in letteratura algoritmi in grado di generare una traiettoria per droni dotati di un carico sospeso [28] oppure l'applicazione di tecniche di controllo predittivo per raggiungere obiettivi di *pick-and-place* [29]. Negli ultimi anni infatti, grazie allo straordinario incremento delle prestazioni computazionali, la generazione di traiettoria tramite controllo predittivo si è aperta la strada anche nell'ambito del controllo dei sistemi meccatronici. La generazione della traiettoria online conferisce infatti il vantaggio di tenere conto dell'effettiva evoluzione del

Figura 1.3: Afferraggio *avian-inspired*.

sistema [30].

Si deve infine sottolineare come i manipolatori aerei siano per loro natura robot ridondanti dal punto di vista del posizionamento dell'*end-effector*. Ciò permette a questi sistemi di agire con grande destrezza e flessibilità nei confronti dell'ambiente nel quale sono immersi. D'altro canto la proprietà di ridondanza pone dei problemi nella risoluzione della cinematica inversa del sistema poichè rende infinite le soluzioni possibili. Per risolvere il sistema in forma chiusa si possono introdurre compiti addizionali estendendo la dimensione dello spazio di lavoro tramite vincoli funzione delle variabili di giunto [31]. Tale espansione può però generare conflitti fra gli obiettivi che devono essere raggiunti, dando luogo al non raggiungimento di nessuno dei *task* da parte del sistema. Per superare questi problemi un approccio può essere l'applicazione della *task priority strategy*. Questa strategia permette di definire una rigida priorità di obiettivi, ottimizzando il primo e trovando una soluzione per gli obiettivi secondari tale da non influenzare la soluzione degli obiettivi a priorità superiore. Ciò viene raggiunto sfruttando tecniche di proiezione nello spazio nullo di ciascun *task* [32], [33].

Nell'ambito della manipolazione aerea lo sfruttamento efficace della ridondanza del sistema è tutt'ora un'ambito di ricerca molto attuale [34], [35]. In questo ambito la ridondanza permette di ottimizzare obiettivi

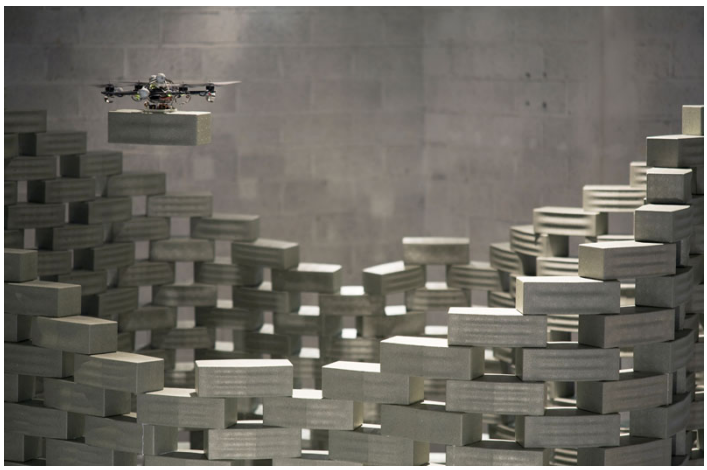


Figura 1.4: Applicazioni di costruzione.

secondari come la riduzione del disallineamento tra baricentro del braccio e del drone, mentre viene raggiunto un obiettivo primario (inseguimento di traiettoria o asservimento visivo).

1.2 Struttura della Tesi

Nel Capitolo 2 viene descritto il sistema sotto studio. In particolare viene presentata sia una descrizione tecnica dei componenti, dei sensori e degli attuatori utilizzati, sia una descrizione del modello matematico che descrive la fisica del sistema.

L'algoritmo utilizzato per raggiungere l'obiettivo del controllo viene presentato nel Capitolo 3. In particolare, nella prima parte si espone la trattazione teoria del controllo predittivo lineare e non lineare, e l'introduzione di un obiettivo secondario implementato gerarchicamente. Nella seconda parte si presenta l'applicazione dell'algoritmo al controllo del sistema in esame, con i relativi modelli e vincoli utilizzati.

Nel Capitolo 4 vengono presentati i risultati e la validazione delle simulazioni nelle quali è stato implementato l'algoritmo di controllo sintetizzato. In particolare, nella prima parte si presentano i risultati delle simulazioni condotte in ambiente Matlab, mentre nella seconda parte si presentano quelli relativi alle simulazioni condotte in ambiente Gazebo.

Infine si presentano nel Capitolo 5 i risultati delle prove sperimentali che sono state condotte allo scopo di effettuare la stima di alcuni parametri del sistema reale e validare il controllo di basso livello.

Capitolo 2

Appartato Sperimentale e Modello Matematico

Una descrizione dettagliata del sistema sul quale si implementeranno le logiche di controllo è fondamentale per comprendere il percorso che è stato seguito durante la realizzazione dei modelli e dei controllori. In questa sezione si presentano quindi le caratteristiche tecniche del sistema e la descrizione del modello matematico, con particolare attenzione all'integrazione dei due principali componenti del sistema complessivo, ovvero drone e braccio. Si presentano inoltre i componenti secondari necessari per il controllo del sistema, quali schede elettroniche, sensori e architettura di comunicazione.

2.1 Drone

Il componente fondamentale costituente il sistema è l'ottocottero. Esso costituisce la base mobile su cui si innesta il braccio robotico, permettendo l'incremento di mobilità e destrezza del manipolatore. L'ottocottero è una tipologia di drone poco diffusa. Nel nostro caso è costituito da un corpo centrale che supporta l'elettronica necessaria per il volo (schede di controllo, batterie, sensori ecc.) e da quattro bracci disposti a croce alle cui estremità sono collocate le eliche azionate per mezzo di motori *brushless*. A differenza dei tradizionali quadricotteri dotati di solo 4 motori,

Figura 2.1: Ottocottero X8+.

sono presenti altri 4 motori disposti in maniera coassiale rispetto ai primi. La presenza di 4 motori addizionali rispetto ai 4 tradizionali è funzionale all'incremento della spinta verticale, necessaria per il sollevamento del peso aggiuntivo del braccio. L'attuazione delle eliche permette la generazione della forza per sostenere e controllare il moto del velivolo nello spazio. Ciascuna elica permette infatti la generazione di forze perpendicolari (forze di *lift*) e parallele (forze di *drag*) al proprio piano di rotazione. Si generano in questo modo delle coppie sul corpo centrale che permettono il controllo dell'orientamento del drone. Oltre alle coppie, la somma delle forze genera una risultante diversa da zero, che permette il sollevamento del corpo e il raggiungimento della condizione di *hovering*, ovvero lo stazionamento del drone in volo. Si nota che pur avendo il controllo sulla velocità di otto motori, tali variabili di controllo si manifestano dal punto di vista della dinamica sotto forma di tre coppie ed una forza ovvero quattro variabili di controllo "dinamiche". Considerando dunque il sistema dinamico contenente posizione e orientamento del sistema, è immediato giungere alla conclusione che si tratti di un sistema "sotto-attuato".

Entrando nel dettaglio del sistema utilizzato, ci siamo orientati sull'utilizzo dell'ottocottero X8+ [36], prodotto da 3DR, principalmente perchè in grado di trasportare un carico utile sufficiente al supporto del manipolatore ed equipaggiato con una scheda Pixhawk autopilot, contenente l'elettronica di controllo. Quest'ultima è particolarmente adatta alle nostre esigenze in quanto *open-source* dal punto di vista software e facilmente interfacciabile ad altre periferiche dal punto di vista hardware, oltre che già equipaggiata con i principali sensori necessari per la stabilizzazione e il volo del drone presentati in sez. 2.3. Oltre ai sensori presenti sulla scheda Pixhawk, sono presenti alcuni sensori aggiuntivi che permettono una migliore stima dello stato del sistema:

- Optical Flow: dotato di sonar, in grado di stimare le velocità traslazionali del drone [51].
- Telecamera Microsoft HD 5000: in grado di stimare la posizione del drone tramite analisi di immagine.

Infine per quanto riguarda gli attuatori, il drone è dotato di 8 motori SunnySky V2216-12 KV800 controllati tramite regolatori di velocità (ESC[1]) che sopportano una corrente massima di 20A. Sui motori vengono montate eliche di dimensione 11x4.7.

2.2 Manipolatore

Al di sotto della base mobile è presente il braccio robotico, ovvero l'elemento di manipolazione vera e propria. Il manipolatore è costuituito da un insieme di corpi rigidi (*link*) tra loro connessi tramite giunti (*joint*) che si classificano in base al moto che essi permettono: giunti prismatici se il moto permesso è di traslazione, giunti rotazionali se il moto permesso è appunto di rotazione. La struttura cinematica che si genera connettendo tra loro i diversi *link* in modo almeno un *link* sia dotato di un solo accoppiamento viene chiamata catena cinematica aperta. I manipolatori possono essere

[1]Electronic Speed Control = controllo elettronico di velocità

classificati in funzione della tipologia e alla successione di giunti utilizzati, quello da noi utilizzato è di tipo antropomorfo. Tale tipologia è stata scelta in quanto grazie alla presenza di giunti solo rotazionali permette il raggiungimento di livelli di destrezza più elevati.

Figura 2.2: CrustCrawler.

Scendendo nel dettaglio della scelta del nostro manipolatore ci siamo orientati sul modello CrustCrawler [37], caratterizzato da quattro giunti rotazionali che conferiscono quindi quattro gradi di libertà. L'end-effector è costituito da una pinza per permettere appunto la manipolazione e l'interazione con l'ambiente. Ciascun giunto giunto viene controlato con un servomotori Dynamixel funzionanti in corrente continua. Questi sono forniti con un microcontrollore integrato che si occupa di controllare tutte le funzioni del motore e di monitorarne lo stato. Ciascun giunto è inoltre fornito di un encoder per misurarne la posizione angolare. Se ne riportano le specifiche tecniche in tabella: 2.1.

2.3 Microcontrollore

L'ottocottero è equipaggiato con una scheda di controllo Pixhawk autopilot, riportata in fig. 2.3, dotata dei principali sensori necessari alla stabilizzazione e al controllo del drone, riportati in seguito. Tale scheda è

Peso	54.6g
Dimensioni	32 x 50.1 x 40 mm
Rapporto di riduzione	1:254
Tensione operativa	9 - 12 V
Corrente di Stand-By	50 mA
Coppia massima	1.5Nm @ 12 V / 1.5A
Massima velocità a vuoto	59RPM @ 12V
Interfaccia di comunicazione	TTL Half Duplex
Sensore di posizione	Potenziometro
Risoluzione	300° (1024 bit)
Motore	Coreless

Tabella 2.1: Specifiche tecniche dei servocomandi Dynamixel

inoltre accompagnata dal software open-source PX4 sviluppato da diversi centri di ricerca [38], e facilmente modificabile [39]. Tale scheda è infine caratterizzata da un'elevata flessibilità e possibilità di interfacciarsi con hardware provenienti da diverse piattaforme. Ne segue che è ottima per le nostre esigenze di comunicazione tra la piattaforma mobile e il braccio robotico.

Figura 2.3: Pixhawk.

Nel dettaglio le caratteristiche tecniche:

- Processore: CPU Cortex-M4F a 32bit e 168 MHz, con 256 KB RAM e 2 MB di memoria Flash.
- 14 uscite PWM per interfacciare la scheda con eventuali attuatori aggiuntivi.
- bus I2C, UART e CAN per collegare sensori esterni.

- Ingresso SD card per salvare i dati di volo.
- Porta dedicata per buzzer per segnalare lo stato del sistema.

Inoltre la scheda monta al suo interno alcuni sensori fondamentali per il volo del drone:

- Giroscopio a tre assi a 16 bit.
- Accelerometro a tre assi.
- Magnetometro.
- Barometro.
- GPS.

A causa dell'elevato carico computazionale del sistema complessivo e alla necessità di interfacciarsi con i diversi dispositivi in maniera più efficiente, il sistema è equipaggiato con una seconda unità di calcolo che permette di assolvere i compiti di analisi di immagine, e generazione di traiettorie.

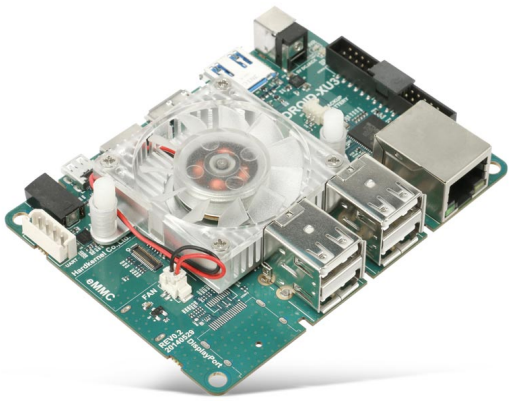


Figura 2.4: Odroid XU3

La scheda scelta è quella riportata in figura 2.4. Si tratta di una Odroid XU3, che presenta elevate prestazioni computazionali [40]. Le caratteristiche tecniche sono qui riportate:

- due CPU quad core: una Samsung Exynos5422 Cortex^{TM}-A15 2.0Ghz e una Cortex^{TM}-A7
- 2 GB di memoria RAM a 933MHz
- una memoria Flash eMMC5.0 HS400
- quattro porte USB 2.0, una 3.0 e una 3.0 OTG
- ingresso HDMI e DisplayPort

Dal punto di vista software la scheda è dotata di Linux Ubuntu 14.04 e ROS (Robot Operating System), ovvero un set di librerie software e tools che ci permettono l'implementazione e l'agevole comunicazioni tra le diverse piattaforme presenti sul sistema [41].

2.4 Architettura di comunicazione

Il sistema complessivo è quindi composto da componenti che devono essere messi in comunicazione tra loro utilizzando diverse piattaforme di comunicazione. La scheda Odroid e il computer remoto utilizzano il software ROS (Robot Operating System) descritto nel dettaglio nella sezione 4.3.1. Per la comunicazione tra ROS e la scheda di controllo Pixhawk si utilizza invece il protocollo MAVlink che gestisce i messaggi tramite l'applicazione MAVROS. Le comunicazioni all'interno della scheda Pixhawk vengono gestite tramite l'applicazione uORB.

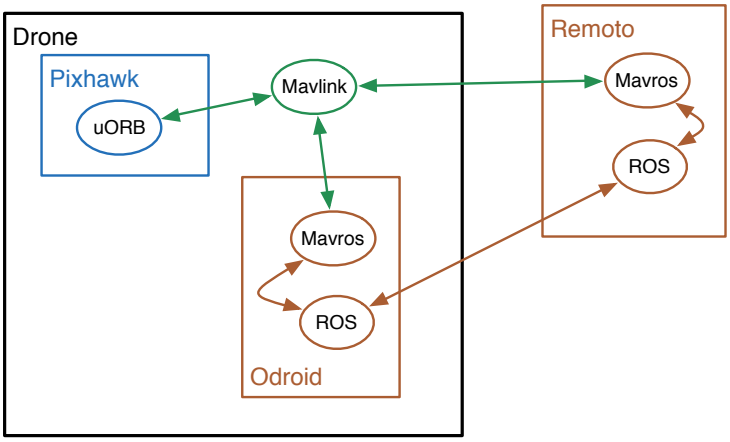


Figura 2.5: Schema di comunicazione tra i diversi componenti

Infine le connessioni sono realizzate tra i diversi componenti tramite porte seriale o porta USB tra le componenti sul drone. Durante il volo la piattaforma è in comunicazione da remoto tramite connessione Wireless.

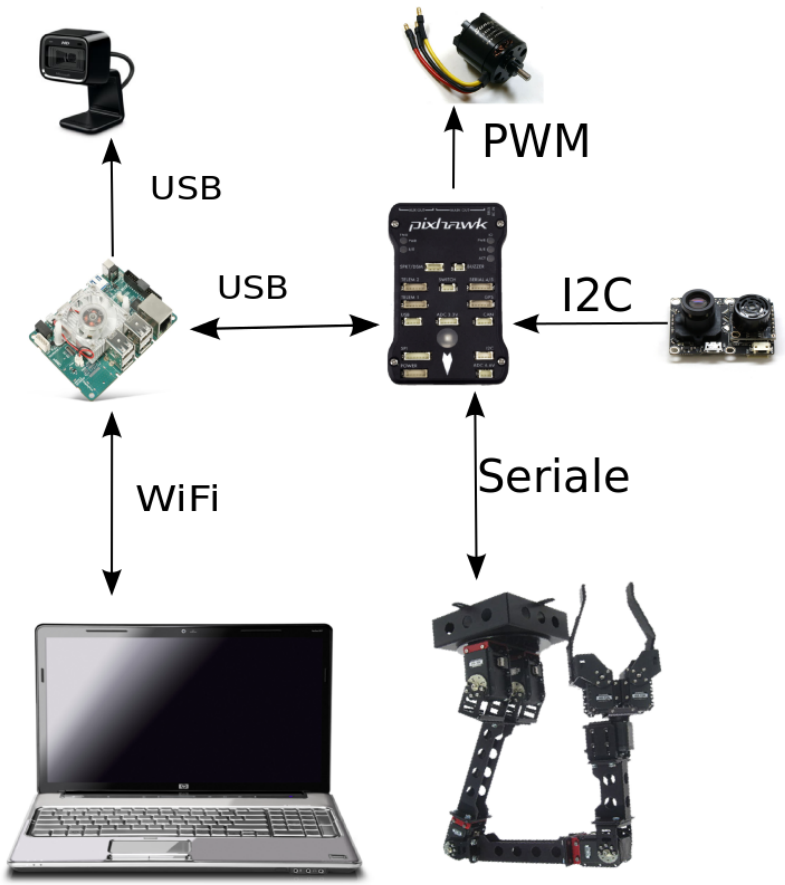


Figura 2.6: Comunicazione tra le componenti

2.5 Cinematica

Per la corretta comprensione dell'algoritmo di controllo e del sistema complessivo è necessario descrivere in maniera dettagliata la cinematica. Innanzitutto è necessario chiarire la disposizione dei diversi sistemi di riferimento disposti sul robot (fig. 2.7). Verranno utilizzate tre terne principali: un sistema di riferimento inerziale (E_i), un sistema di riferimento posto sul quadricottero (E_b) un sistema di riferimento posto sull'end effector del manipolatore (E_e).

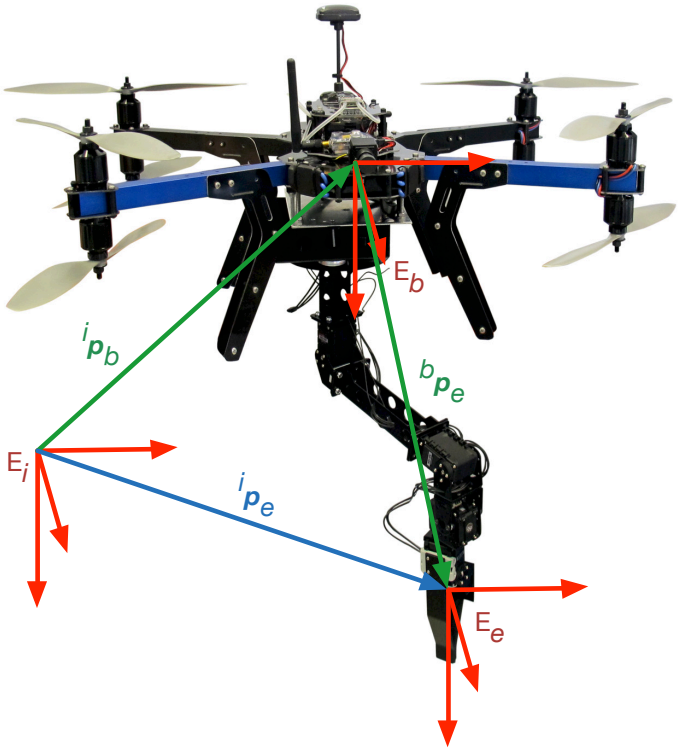


Figura 2.7: Disposizione delle terne di riferimento.

Per quanto riguarda la cinematica della base valgono le classiche relazioni valide per ogni veicolo aereo:

$$\begin{aligned} {}^{i}\dot{\mathbf{p}}_b &= {}^{i}R_b\,{}^{b}\mathbf{v}_b \\ {}^{i}\dot{\boldsymbol{\phi}}_b &= {}^{i}\Gamma_b\,{}^{b}\boldsymbol{\omega}_b \end{aligned} \tag{2.1}$$

dove ${}^i\mathbf{p}_b = [x, y, z]^T$ è la posizione assoluta del centro della piattaforma espresso in E_i e ${}^i\boldsymbol{\phi}_b = [\psi, \vartheta, \varphi]^T$ il suo orientamento assoluto espresso tramite gli angoli di Eulero, ${}^b\mathbf{v}_b$ è la velocità della piattaforma espressa in E_b e ${}^b\boldsymbol{\omega}_b$ è il vettore che contiene le velocità angolari del corpo in E_b. ${}^iR_b \in SO(3)$ è infine la matrice di rotazione:

$$ {}^iR_b = \begin{bmatrix} c_\psi c_\vartheta & c_\psi s_\vartheta s_\varphi - s_\psi c_\varphi & s_\psi s_\varphi + c_\psi s_\vartheta c_\varphi \\ s_\psi c_\vartheta & c_\psi c_\varphi + s_\psi s_\vartheta s_\varphi & s_\psi s_\vartheta c_\varphi - c_\psi s_\varphi \\ -s_\vartheta & c_\vartheta s_\varphi & c_\vartheta c_\varphi \end{bmatrix} \tag{2.2} $$

mentre la matrice ${}^i\Gamma_b$ è la matrice di trasferimento:

$$ {}^i\Gamma_b = \begin{bmatrix} 1 & s_\varphi t_\vartheta & c_\varphi t_\vartheta \\ 0 & c_\varphi & -s_\varphi \\ 0 & s_\varphi / c_\vartheta & c_\varphi / c_\vartheta \end{bmatrix} . \tag{2.3} $$

Per quanto riguarda invece la cinematica del robot, essa viene studiata definendo innanzitutto i parametri di Denavit-Hartenberg riportati in tab. 2.2 e ricavando le matrici di trasfomazione omogenea necessarie per la costruzione dello Jacobiano [42].

Tabella 2.2: Parametri Denavit-Hartemberg per il manipolatore robotico a 4 GdL

Joint	d [mm]	ϑ	a [mm]	α [rad]
1	d_1	ϑ_1	0	$-\pi/2$
2	0	ϑ_2	a_2	0
3	0	ϑ_3	0	$-\pi/2$
4	d_4	ϑ_4	0	0

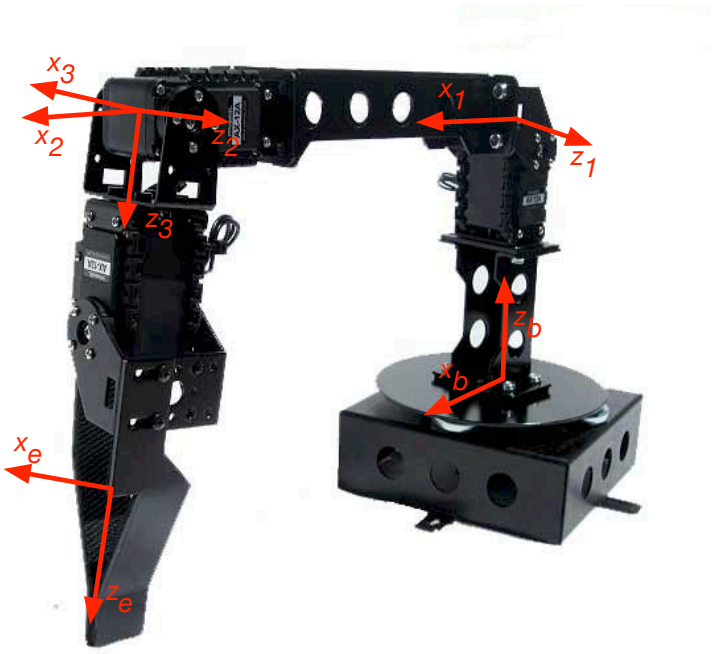


Figura 2.8: Disposizione delle terne del braccio robotico.

Le matrici ottenute sono:

$$ {}^{b}A_1(\vartheta_1) = \begin{bmatrix} \sin(\vartheta_1) & 0 & -\sin(\vartheta_1) & 0 \\ \sin(\vartheta_1) & 0 & \cos(\vartheta_1) & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.4} $$

$$ {}^{1}A_2(\vartheta_2) = \begin{bmatrix} \cos(\vartheta_2) & -\sin(\vartheta_2) & 0 & a_2\cos(\vartheta_2) \\ \sin(\vartheta_2) & \cos(\vartheta_2) & 0 & a_2\sin(\vartheta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.5} $$

$$ {}^{2}A_3(\vartheta_3) = \begin{bmatrix} \cos(\vartheta_3) & 0 & -\sin(\vartheta_3) & 0 \\ \sin(\vartheta_3) & 0 & \cos(\vartheta_3) & 0 \\ 0 & -1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.6} $$

$$
{}^{3}A_e(\vartheta_4) \;=\; \begin{bmatrix} \cos(\vartheta_4) & -\sin(\vartheta_4) & 0 & 0 \\ \sin(\vartheta_4) & \cos(\vartheta_4) & 0 & 0 \\ 0 & 0 & 1 & d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{2.7}
$$

A questo punto è possibile ricavare la posizione dell'*end-effector* rispetto a E_b, selezionando le prime tre righe dell'ultima colonna dal prodotto delle diverse matrici:

$$
{}^{b}A_e(\vartheta_1, \vartheta_2, \vartheta_3, \vartheta_4) = {}^{b}A_1\,{}^{1}A_2\,{}^{2}A_3\,{}^{3}A_e \tag{2.8}
$$

$$
{}^{b}\mathbf{p}_e = \begin{bmatrix} a_2\cos(\vartheta_1)\cos(\vartheta_2) - d_4(\cos(\vartheta_1)\cos(\vartheta_2)\sin(\vartheta_3) + \cos(\vartheta_1)\sin(\vartheta_2)\cos(\vartheta_3)) \\ a_2\sin(\vartheta_1)\cos(\vartheta_2) - d_4(\sin(\vartheta_1)\cos(\vartheta_2)\sin(\vartheta_3) + \sin(\vartheta_1)\sin(\vartheta_2)\cos(\vartheta_3)) \\ d_1 - d_4(\cos(\vartheta_2)\cos(\vartheta_3) - \sin(\vartheta_2)\sin(\vartheta_3)) - a_2\sin(\vartheta_2) \end{bmatrix} \tag{2.9}
$$

Il vettore ${}^{b}\mathbf{p}_e$ è la posizione dell'*end-effector* rispetto alla terna E_b in funzione delle variabili di giunto. Seguendo lo stesso procedimento è possibile ricavare le posizioni dei diversi giunti del braccio robotico ${}^{b}\mathbf{p}_1$, ${}^{b}\mathbf{p}_2$, ${}^{b}\mathbf{p}_3$. Nota la posizione di ciascun giunto del braccio robotico in funzione delle variabili di giunto, è immediato calcolare la posizione del baricentro complessivo dello stesso. Infatti, supponendo che il baricentro di ciascun *link* sia posto nel centro dello stesso, è immediato calcolare la posizione di ciascuno rispetto alla terna E_b:

$$
{}^{b}\mathbf{p}_{G,1} = ({}^{b}\mathbf{p}_1)/2 \tag{2.10}
$$

$$
{}^{b}\mathbf{p}_{G,2} = ({}^{b}\mathbf{p}_2 + {}^{b}\mathbf{p}_3)/2 \tag{2.11}
$$

$$
{}^{b}\mathbf{p}_{G,3} = ({}^{b}\mathbf{p}_3 + {}^{b}\mathbf{p}_e)/2\,. \tag{2.12}
$$

Nota la posizione del baricentro di ciascun *link*, è possibile effettuare una media pesata sulla massa di ciascun *link* (m_j) per trovare la posizione del baricentro del braccio robotico:

$$
{}^{b}\mathbf{p}_G \;=\; \frac{(\sum_{j=1}^{N_l} m_j\,\mathbf{p}_{G_{j,i}})}{\sum_{j=1}^{N_l} m_j}\,. \tag{2.13}
$$

L'ultimo passaggio necessario per il calcolo della velocità dell'*end-effector* è ricavare lo Jacobiano del robot. Per fare ciò è necessario ricavare gli assi $\mathbf{z_i}$ di ogni terna i-esima, definiti dalle prime tre componenti della terza colonna di ciascuna matrice ${}^b\mathbf{A}_i$. Riportiamo qui solo il vettore $\mathbf{z}_e$:

$$\mathbf{z}_e = \begin{bmatrix} -\cos(\vartheta_1)\cos(\vartheta_2)\sin(\vartheta_3) - \cos(\vartheta_1)\sin(theta_2)\cos(\vartheta_3) \\ -\sin(\vartheta_1)\cos(\vartheta_2)\sin(\vartheta_3) - \sin(\vartheta_1)\sin(\vartheta_2)\cos(\vartheta_3) \\ \sin(\vartheta_2)\sin(\vartheta_3) - \cos(\vartheta_2)\cos(theta_3) \end{bmatrix}. \tag{2.14}$$

Noti dunque gli assi z_i e le posizioni ${}^b\mathbf{p}_i$ di ogni giunto, ed essendo ognuno di questi di tipo rotoidale, è possibile ricavare lo Jacobiano scrivendo l'i-esima colonna come:

$$\begin{bmatrix} \mathbf{J}_{Pi} \\ \mathbf{J}_{Oi} \end{bmatrix} = \begin{bmatrix} \mathbf{z}_{i-1} \times ({}^b\mathbf{p}_e - {}^b\,\mathbf{p}_{i-1}) \\ \mathbf{z}_{i-1} \end{bmatrix}. \tag{2.15}$$

Accostando le i-esime colonne $\mathbf{J}_{Pi}$ si ricava la matrice ${}^bJ_m \in \mathbb{R}^{3\times4}$ che esprime, in funzione della particolare configurazione del manipolatore, il legame tra le velocità lineari e le derivate temporali delle variabili di giunto:

$${}^b\mathbf{v}_e = {}^bJ_m\dot{\mathbf{q}}. \tag{2.16}$$

A questo punto concentriamo la nostra attenzione su come la base mobile influenza la cinematica del nostro robot. Poichè il drone costituisce la base mobile del braccio robotico, la posizione dell'*end-effector* sarà pari alla somma dei vettori descriventi la posizione della base rispetto alla terna inerziale e la posizione dell'*end-effector* nella terna base, ruotata opportunamente. Scritto in termini matematici:

$${}^i\mathbf{p}_e = {}^i\mathbf{p}_b + {}^iR_b\,{}^b\mathbf{p}_e\,, \tag{2.17}$$

dove ${}^i\mathbf{p}_e$ è la posizione dell'*end-effector* in E_i e ${}^b\mathbf{p}_e$ è la posizione dell'*end-effector* in E_b. Nota la posizione dell'*end-effector* è possibile ricavare la velocità dello stesso derivando la relazione rispetto al tempo:

$$ {}^{i}\dot{\mathbf{p}}_e = {}^{i}\dot{\mathbf{p}}_b + {}^{i}\dot{R}_b\, {}^{b}\mathbf{p}_e + {}^{i}R_b\, {}^{b}\dot{\mathbf{p}}_e \,, \tag{2.18} $$

dove il termine contenente la derivata della matrice di rotazione può essere riscritto come:

$$ {}^{i}\dot{R}_b\, {}^{b}\mathbf{p}_e = {}^{i}R_b\, S({}^{b}\boldsymbol{\omega}_b)\, {}^{b}\mathbf{p}_e \tag{2.19} $$

e la matrice $\mathbf{S}(\cdot)$ è una matrice antisimmetrica funzione delle velocità angolari del drone in E_b. In definitiva la velocità dell'*end-effector* può essere riscritta come:

$$ {}^{i}\dot{\mathbf{p}}_e = {}^{i}\dot{\mathbf{p}}_b + {}^{i}R_b\, S({}^{b}\boldsymbol{\omega}_b)\, {}^{b}\mathbf{p}_e + {}^{i}R_b\, {}^{b}J_m\dot{\mathbf{q}} \tag{2.20} $$

Notiamo come il sistema complessivo sia dotato nel complesso di sei GdL per quanto riguarda la base mobile (tre traslazionali e tre rotazionali) e quattro GdL per quanto riguarda il robot, per un totale di 10 GdL. Dobbiamo però osservare che in realtà la base mobile permette l'attuazione diretta di solo quattro GdL, e gli angoli di Eulero di rollio e beccheggio (*roll* e *pitch*) dovranno essere asserviti alla stabilizzazione del drone. In definitiva il sistema è dotato di otto GdL attuabili, quindi ridondante dal punto di vista del posizionamento dell'*end-effector*.

2.6 Dinamica

Dopo aver descritto la cinematica del sistema ci occupiamo del comportamento dinamico dello stesso, aspetto fondamentale per la successiva comprensione dei sistemi di controllo e dei modelli utilizzati. La sezione è organizzata descrivendo la dinamica di ciascuno del principali elementi costituenti il sistema. Innanzitutto viene presentata la dinamica del drone, con particolare attenzione alla aerodinamica. In secondo luogo si descrive la dinamica del braccio robotico. Infine si ricava il comportamento dinamico complessivo.

2.6.1 Drone

In primo luogo introduciamo l'approssimazione a quattro eliche del nostro drone, ovvero, pur utilizzando otto eliche a due a due coassiali nei nostri modelli, caratterizziamo l'ottocottero come un semplice quadricottero, in grado quindi di generare quattro forze in prossimità degli assi dei motori. Introdotta questa semplificazione è sufficiente seguire la descrizione dinamica dei classici quadricotteri. Seguendo l'approccio di Newton-Eulero è possibile ricavare il modello dinamico del sistema drone ([43]):

$$\begin{aligned} {}^{i}\ddot{\mathbf{p}}_b &= -\mathbf{g} + I_T^{-1}\, {}^{i}R_b\, \mathbf{T} \\ {}^{b}\dot{\boldsymbol{\omega}}_b &= I_R^{-1}\,(-{}^{b}\boldsymbol{\omega}_b \times I_R\, {}^{b}\boldsymbol{\omega}_b) + I_R^{-1}\,\boldsymbol{\tau}\,, \end{aligned} \tag{2.21}$$

dove ${}^{i}\ddot{\mathbf{p}}_b$ sono le accelerazioni della piattaforma mobile in E_i, ${}^{i}R_b$ è la matrice di rotazione, ${}^{b}\dot{\boldsymbol{\omega}}_b$ è la derivata delle velocità angolari del corpo in terna E_b; $\mathbf{g}$ indica il vettore di gravità costante, I_T e I_R sono rispettivamente le matrici di inerzia traslazionale e rotazionale solidali con il drone, e $\mathbf{T}$ e $\boldsymbol{\tau}$ corrispondono a forza di Thrust e coppie agenti sul telaio del drone.

Aerodinamica

Per la comprensione di come è possibile generare le forze per il controllo del sistema è necessario descrivere in parte l'aerodinamica. L'aerodinamica dei rotori è stata esaustivamente studiata nel corso del secolo scorso, e

dettagliati modelli di ogni tipo di elica sono presenti in letteratura [44]. La forza verticale generata da un'elica posta in rotazione attorno al proprio asse, a velocità costante nell'aria libera, può essere modellata come:

$$T_i = C_T \rho A_{r_i} r_i^2 \omega_i^2 \tag{2.22}$$

dove per l'i-esima elica, A_{r_i} è l'area del disco formato dall'elica nel suo movimento, r_i è il raggio dell'elica stessa, ω_i è la velocità di rotazione della stessa, C_T è un coefficiente costante che dipende dalla particolare elica (coefficiente di *Thrust*) e infine ρ è la densità dell'aria. In pratica, la caratterizzazione dell'elica tramite prove sperimentali permette di ricavare il coefficiente c_T:

$$T_i = c_T \omega_i^2 \tag{2.23}$$

Tale caratterizzazione sperimentale permette inoltre di ricavare anche la coppia di reazione derivante dall'effetto di *Drag*, che segue la stessa relazione matematica:

$$Q_i = c_Q \omega_i^2 \tag{2.24}$$

Si suppone come prima approssimazione che l'asse di rotazione delle eliche sia sempre orientato come l'asse z di E_b, anche se in realtà ciò non è sempre vero, infatti nel momento in qui il sistema inizia a ruotare e muoversi attraverso l'aria dà luogo a fenomeni aerodinamici più complessi (*Rotor Flapping*). La forza verticale totale generata dalle eliche sarà dunque pari alla somma di ciascuna forza:

$$T = \sum_{i=1}^{N} |T_i| \tag{2.25}$$

Per quanto riguarda invece le coppie generate sul telaio del drone esse dipenderanno dalla forza generata dalle eliche, ma anche dalla geometria del sistema stesso. Se associamo a ciascun i-esima elica un angolo ϕ_i tra il braccio di lunghezza d su cui essa è posta e l'asse x di E_b, possiamo scrivere:

$$\begin{aligned} \tau_1 &= \sum_{i=1}^{N} d_i \sin(\phi_i)|T_i| \\ \tau_2 &= \sum_{i=1}^{N} d_i \cos(\phi_i)|T_i| \\ \tau_3 &= \sum_{i=1}^{N} d_i \sigma_i |Q_i| \end{aligned} \tag{2.26}$$

dove σ_i dipende dal verso di rotazione dell'elica stessa. Per un generico quadricottero queste relazioni possono essere scritte in forma matriciale:

$$\begin{bmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} c_T & c_T & c_T & c_T \\ 0 & dc_T & 0 & -dc_T \\ -dc_T & 0 & dc_T & 0 \\ -c_Q & c_Q & -c_Q & c_Q \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix} \tag{2.27}$$

Dove la matrice descrivente il sistema può essere chiamata genericamente M. Il sistema lineare così ottenuto è facilmente invertibile e ci permette di ricavare i valori di velocità da assegnare a ciascun motore per ottenere la sequenza di forze/coppie richiesta (operazione di *mixing*). Nel nostro

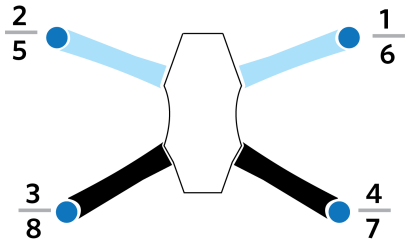


Figura 2.9: Posizione dei Motori

caso l'implementazione sul robot reale richiederà la costruzione di una matrice più ampia a causa della presenza di quattro motori addizionali, quindi quattro colonne in più. Nota la posizione dei motori (figura 2.9) e le direzioni di rotazione, la matrice assume la forma:

$$\begin{bmatrix} T \\ \tau_1 \\ \tau_2 \\ \tau_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ ds_\alpha & -ds_\alpha & -ds_\alpha & ds_\alpha & -ds_\alpha & ds_\alpha & ds_\alpha & -ds_\alpha \\ -dc_\alpha & -dc_\alpha & dc_\alpha & dc_\alpha & -dc_\alpha & -dc_\alpha & dc_\alpha & dc_\alpha \\ c & -c & c & -c & c & -c & c & -c \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ ... \\ T_8 \end{bmatrix} \tag{2.28}$$

dove:

- d è la distanza tra il motore e il centro del velivolo
- s_α e c_α rappresentano rispettivamente il seno e il coseno dell'angolo tra l'asse di simmetria frontale e i bracci anteriori;
- c è un coefficiente che rappresenta il rapporto tra i coefficienti di *Thrust* e *Drag* delle eliche.

Noti i valori di *Thrust* e coppie richieste è possibile ricavare le forze che devono essere generate da ciascun motore (da convertire poi in velocità equivalenti delle eliche) utilizzando la pseudo inversa della matrice M del nostro sistema. L'utilizzo della pseudo inversa della matrice $M^\dagger$:

$$M^\dagger = M^T(MM^T)^{-1} \tag{2.29}$$

garantisce che la soluzione del problema ottenuto sia di minima norma.

2.6.2 Braccio robotico

La dinamica del *CrastCrawler* viene studiata a partire dal classico modello dei manipolatori robotici [42]:

$$B(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau}_m \,. \tag{2.30}$$

dove:

- $B(\mathbf{q})$ è la matrice di inerzia del robot dipendente dalla configurazione del robot stesso;
- $C(\mathbf{q}, \dot{\mathbf{q}})$ è la matrice contenente i termini centrifughi e di Coriolis;
- $\mathbf{g}(\mathbf{q})$ è il vettore contenente i termini grazitazionali;
- $\boldsymbol{\tau}_m$ sono le coppie di giunto.

2.6.3 Sistema complessivo

La dinamica del sistema complessivo dovrà tenere conto delle interazioni dinamiche tra quadricottero e braccio robotico. Per ricavare le equazioni

ricorriamo all'approccio di Eulero-Lagrange, basato sul principio di conservazione dell'energia meccanica. Consideriamo la classica equazione di conservazione:

$$L = K - P\,. \tag{2.31}$$

dove l'energia cinetica K è composta dai contributi di moto della base (K_b) e di ogni *link* del braccio (K_{q_i}):

$$K = K_b + \sum_{i=1}^{n} K_{q_i}\,. \tag{2.32}$$

Nel caso di un veicolo aereo:

$$K_b = \frac{1}{2} m_b {}^i\dot{\mathbf{p}}_b^T {}^i\dot{\mathbf{p}}_b + \frac{1}{2} {}^b\boldsymbol{\omega}_b^T I_R {}^b\boldsymbol{\omega}_b. \tag{2.33}$$

dove m_b e I_R sono rispettivamente la massa e la matrice di inerzia del veicolo, e combinando Eq. 2.33 con Eq. 2.1 otteniamo:

$$K_b = \frac{1}{2} m_b {}^i\dot{\mathbf{p}}_b^T {}^i\dot{\mathbf{p}}_b + \frac{1}{2} {}^i\dot{\boldsymbol{\phi}}_b^T {}^i\Gamma_b^{-T} I_R {}^i\Gamma_b^{-1} {}^i\dot{\boldsymbol{\phi}}_b\,. \tag{2.34}$$

Possiamo definire nello stesso modo l'energia cinetica di ciascun giunto come:

$$K_{q_i} = \frac{1}{2} m_{q_i} {}^i\dot{\mathbf{p}}_{q_i}^T {}^i\dot{\mathbf{p}}_{q_i} + \frac{1}{2} \boldsymbol{\omega}_{q_i}^T {}^iR_{q_i} I_{q_i} ({}^iR_{q_i})^T \boldsymbol{\omega}_{q_i}. \tag{2.35}$$

con m_{q_i}, $\boldsymbol{\omega}_{q_i}$ e I_{q_i} la rispettiva massa, velocità angolare e matrice d'inerzia del i-esimo *link*; e ${}^iR_{q_i} = {}^iR_b {}^bR_{q_i}$, dove ${}^bR_{q_i}$ descrive la rotazione tra l'i-esimo baricentro e il telaio del quadricottero.

La generica j-esima forza non conservativa può essere definita come:

$$\boldsymbol{\Gamma}_j = \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\xi}_j}\right) - \frac{\partial L}{\partial \xi_j}\,, \tag{2.36}$$

con ξ_j la j-esima coordinata generalizzata $\boldsymbol{\xi} = \begin{bmatrix} {}^i\mathbf{p}_b & {}^i\boldsymbol{\phi}_b & \mathbf{q} \end{bmatrix}^T$ e $\dot{\xi}_j$ la sua derivata temporale. Ora combinando i corrispondenti termini dell'Eq. 2.31 in Eq. 2.36 e riordinando, possiamo riscrivere l'energia cinetica totale

come:

$$K = \frac{1}{2}\dot{\xi}^T \mathbf{I}(\xi)\,\dot{\xi}\,, \tag{2.37}$$

dove $\mathbf{I}$ è una matrice simmetrica e definita positiva, funzione delle variabili di drone e braccio. Esattamente come l'energia cinetica anche l'energia potenziale è la composizione di termini dipendenti da drone e braccio robotico:

$$P = P_b + \sum_{i=1}^{n} P_{q_i}\,, \tag{2.38}$$

dove P_b e P_{q_i} sono le energie potenziali associate rispettivamente al drone e all'i-esimo *link*. Questa energia potenziale del veicolo è pari a:

$$P_b = m_b\,\mathbf{g}^T\,{}^i\mathbf{p}_b\,, \tag{2.39}$$

con $\mathbf{g}$ vettore di gravità. Il contributo di energia di ogni *link* del manipolatore può essere riscritto come:

$$P_{q_i} \;=\; m_{q_i}\,\mathbf{g}^T\,({}^i\mathbf{p}_b \;+\; {}^iR_b\,{}^b\mathbf{p}_{q_i})\,. \tag{2.40}$$

Poi combinando e riarrangiando le Eq. 2.38, 2.39 e 2.40, l'energia potenziale del sistema complessivo risulta:

$$P = m_b\,\mathbf{g}^T\,{}^i\mathbf{p}_b \;+\; \mathbf{g}^T \sum_{i=1}^{n} m_{q_i}\,({}^i\mathbf{p}_b \;+\; {}^iR_b\,{}^b\mathbf{p}_{q_i})\,. \tag{2.41}$$

Infine, sostituendo Eq. 2.37 e Eq. 2.41 in Eq. 2.31 otteniamo:

$$\mathbf{I}(\xi)\ddot{\xi} \;+\; \mathbf{C}(\xi,\dot{\xi})\dot{\xi} \;+\; \mathbf{G}(\xi) \;=\; \mathbf{u} \;+\; \mathbf{u}_{ext}\,, \tag{2.42}$$

dove $\mathbf{C}(\xi,\dot{\xi})\dot{\xi}$ rappresentano i termini centrifughi e di Coriolis, e $\mathbf{G}(\xi)$ include gli effetti gravitazionali di ogni giunto. Alla destra dell'uguale troviamo $\mathbf{u_{ext}}$ che rappresenta le forze esterne, mentre $\mathbf{u}$ è un vettore contenente le forze di controllo agenti sul nostro sistema, ovvero forza di

Thrust, coppie sul drone, coppie sui giunti:

$$\mathbf{u} = \begin{bmatrix} {}^{i}\mathbf{R}_{q}\mathbf{T} \\ \boldsymbol{\tau} \\ \boldsymbol{\tau}_{m} \end{bmatrix} . \tag{2.43}$$

Si noti che il modello considerato per semplicità non tiene conto della presenza di attriti.

Capitolo 3

Architettura di Controllo

3.1 Introduzione

Si presenta in questo capitolo la strategia e la trattazione teorica del sistema di controllo progettato. Viene innanzitutto presentata l'architettura generale di controllo proposta. Si presentano poi i principi alla base del controllo predittivo, o MPC (Model Predictive Control) [45]. Nella seconda parte del capitolo viene applicato il controllo predittivo al sistema in esame confrontando l'utilizzo di tre diversi modelli per la predizione. Viene poi proposta una soluzione per sfruttare la ridondanza del sistema tramite approccio gerarchico all'interno del controllo predittivo. Infine, vegono presentati i controllori di basso livello.

3.2 Architettura di controllo

Il sistema di controllo che presentiamo è composto da più anelli in cascata, come prassi nell'ambito della stabilizzazione e inseguimento traiettoria dei droni (fig. 3.1). Essenzialmente il sistema di controllo può essere così suddiviso:

- Alto livello: a livello più alto poniamo l'architettura di controllo MPC essendo la parte del controllo più pesante dal punto di vista computazionale. Il controllo MPC utilizzato influenzerà la struttura

degli anelli di basso livello in funzione del modello utilizzato per la predizione, poichè genererà i riferimenti che andranno inseguiti dagli anellli di livello inferiore. Durante lo studio di questo controllore sono stati sviluppati complessivamente tre modelli: un modello contenente l'intera dinamica del sistema, un modello contenente la dinamica del quadricottero e la cinematica del braccio robotico (modello ibrido), un modello puramente cinematico. La struttura delle diverse tipologie di MPC sono presentate nella sez. 3.4.

- Basso livello: sono gli anelli utilizzati per inseguire i riferimenti generati dal controllo MPC.
  - Nel caso di modello totalmente dinamico il controllo di basso livello si occupa dell'inseguimento dei riferimenti di coppia su quadricottero e giunti braccio (3.4.1).
  - Nel caso di modello ibrido il controllo di basso livello si occupa di inseguire i riferimenti di coppia dei motori del drone, mentre viene utilizzato per inseguire i riferimenti di posizione e velocità dei giunti del braccio (3.4.2).
  - Per quanto riguarda il modello puramente cinematico, il controllo di basso livello si occupa di inseguire i riferimenti di posizione generati da MPC cinematico sia per quanto riguarda la posizione del drone nello spazio sia per quanto riguarda il valore delle variabili di giunto del robot (3.4.3).

La descrizione nel dettaglio degli anelli di basso livello è presentata in sez. 3.5.

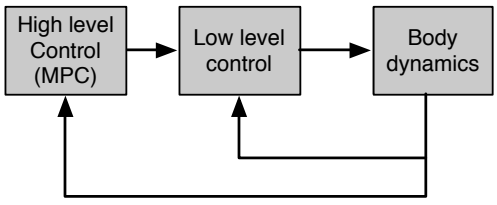


Figura 3.1: Architettura di controllo complessiva.

3.3 Model Predictive Control

Si assuma che il sistema sotto controllo sia descritto dal modello lineare:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)\,,$$

in cui $\mathbf{x} \in \mathbb{R}^n$ è misurabile e $\mathbf{u} \in \mathbb{R}^m$ è il vettore delle variabili di controllo e $A \in \mathbb{R}^{n\times n}$ e $B \in \mathbb{R}^{n\times m}$ sono rispettivamente le matrici di stato e degli ingressi.
All'istante k si vuole determinare la sequenza di variabili di controllo:

$$U(k) = \begin{bmatrix} \mathbf{u}(k) \\ \mathbf{u}(k+1) \\ \vdots \\ \mathbf{u}(k+N-2) \\ \mathbf{u}(k+N-1) \end{bmatrix},$$

che minimizza la cifra di merito quadratica su orizzonte finito:

$$H(\mathbf{x}(k), U(k), k) \;=\; \sum_{i=0}^{N-1} (||\mathbf{x}(k+i)||_Q^2 + ||\mathbf{u}(k+i)||_R^2) + ||\mathbf{x}(k+N)||_S^2\,.^{1}$$

in cui $Q \geq 0 \in \mathbb{R}^{n\times n}$ è una matrice semi-definita positiva usata come matrice dei pesi per lo stato del sistema, $R > 0 \in \mathbb{R}^{m\times m}$ è una matrice definita positiva che definisce i pesi sulle variabili di controllo e $S \geq 0 \in \mathbb{R}^{n\times n}$ è una matrice dei pesi sullo stato finale del sistema. Il numero intero positivo N viene comunemente chiamato orizzonte di predizione. Definito in questo modo, H dipende solo dallo stato attuale e dall'evoluzione dell'attuazione.
La soluzione di questo problema è data dalla legge di controllo ottima e

[1] dove viene utilizzata la notazione abbreviata:

$$||\mathbf{x}||_M^2 = \mathbf{x}^\mathbf{T}\mathbf{M}\mathbf{x} \in \mathbb{R} \tag{3.1}$$

variante nel tempo:

$$\mathbf{u}^o(k+i) = -K(i)\,\mathbf{x}(k+i), \quad i = 0, 1, ..., N-1 \tag{3.2}$$

dove:

$$K(i) = (R + B^T P(i+1)B)^{-1} B^T P(i+1)A\,,$$

e $P(i)$ è la soluzione dell'equazione di Riccati alle differenze:

$$P(i) = Q + A^T P(i+1)A - A^T P(i+1)B(R + B^T P(i+1)B)^{-1} B^T P(i+1)A\,, \tag{3.3}$$

con la condizione al contorno:

$$P(N) = S\,.$$

Si noti che la 3.2 è una legge di controllo in anello chiuso, in quanto la generica variabile di controllo $u(k+i)$ dipende dallo stato corrente $x(k+i)$.

3.3.1 Soluzione in anello aperto

La soluzione del problema di ottimizzazione può essere ricavata a partire dall'equazione di Lagrange:

$$\mathbf{x}(k+1) \;=\; A^i \mathbf{x}(k) + \sum_{j=0}^{i-1} A^{i-j-1}\mathbf{u}(k+j), \quad i > 0,$$

e quindi definendo

$$\mathcal{X}(k) = \begin{bmatrix} \mathbf{x}(k+1) \\ \mathbf{x}(k+2) \\ \vdots \\ \mathbf{x}(k+N-1) \\ \mathbf{x}(k+N) \end{bmatrix}, \; \mathcal{A} = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^{N-1} \\ A^N \end{bmatrix},$$

$$\mathcal{B} = \begin{bmatrix} B & 0 & 0 & \dots & 0 & 0 \\ AB & B & 0 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ A^{N-2}B & A^{N-3}B & A^{N-4}B & \dots & B & 0 \\ A^{N-1}B & A^{N-2}B & A^{N-3}B & \dots & AB & B \end{bmatrix},$$

si ottiene:

$$\mathcal{X}(k) = \mathcal{A}\mathbf{x}(k) + \mathcal{B}U(k)\,. \tag{3.4}$$

Si definiscano infine le matrici con N blocchi sulla diagonale:

$$\mathcal{Q} = \begin{bmatrix} Q & 0 & \dots & 0 & 0 \\ 0 & Q & \dots & 0 & 0 \\ \dots & \dots & \ddots & \dots & \dots \\ 0 & 0 & \dots & Q & 0 \\ 0 & 0 & \dots & 0 & S \end{bmatrix}, \mathcal{R} = \begin{bmatrix} R & 0 & \dots & 0 & 0 \\ 0 & R & \dots & 0 & 0 \\ \dots & \dots & \ddots & \dots & \dots \\ 0 & 0 & \dots & R & 0 \\ 0 & 0 & \dots & 0 & R \end{bmatrix}.$$

Si osservi ora che:

$$\min_{U(k)} H(\mathbf{x}(k), U(k); k) = \min_{U(k)} \bar{H}(\mathbf{x}(k), U(k); k)\,,$$

dove:

$$\bar{H}(\mathbf{x}(k), U(k); k) = \mathcal{X}(k)^T \mathcal{Q}\mathcal{X}(k) + U(k)^T \mathcal{R}U(k)\,,$$

in quanto da $H(\mathbf{x}(k), U(k), k)$ si può eliminare il termine $\mathbf{x}(k)^T Q\mathbf{x}(k)$ che non dipende da $\mathbf{u}(k+j), j \geq 0$. Si può dunque scrivere considerando la eq. 3.4:

$$\bar{H}(\mathbf{x}(k), U(k); k) = (\mathcal{A}\mathbf{x}(k) + \mathcal{B}U(k))^T \mathcal{Q}(\mathcal{A}\mathbf{x}(k) + \mathcal{B}U(k)) + U(k)^T \mathcal{R}U(k)\,,$$

Derivando rispetto a $U(k)$ la formulazione quadratica del problema scritta precedentemente e ponendola uguale a zero si ottiene il vettore degli ingressi ottimo $U^o(k)$ nell'orizzonte considerato:

$$U^o(k) = -(\mathcal{B}^T \mathcal{Q}\mathcal{B} + \mathcal{R})^{-1} \mathcal{B}^T \mathcal{Q}\mathcal{A}\mathbf{x}(k)\,. \tag{3.5}$$

La soluzione così ottenuta dipende dalla predizione del successivo valore dello stato, effettuata in base allo stato corrente $\mathbf{x}(k)$. Infatti si può scrivere:

$$\mathcal{K} = (\mathcal{B}^T\mathcal{Q}\mathcal{B} + \mathcal{R})^{-1}\mathcal{B}^T\mathcal{Q}\mathcal{A} = \begin{bmatrix} \mathcal{K}(0) \\ \vdots \\ \mathcal{K}(N-1) \end{bmatrix}; \quad \mathcal{K}(i) \in \mathbb{R}^{m\times n}\,,$$

quindi

$$\mathbf{u}^o(k+i) = -\mathcal{K}(i)\mathbf{x}(k),\ i = 0, 1, ..., N-1\,. \tag{3.6}$$

Per questo motivo si può interpretare la legge di controllo di eq. 3.6 come una soluzione in anello aperto per $i > 0$. Si noti come nel caso di assenza di disturbi o errori nel modello, soluzione in anelllo aperto ed in anello chiuso coincidano. La differenza sostanziale tra i due approcci riguarda la facilità di introdure vincoli sulle varibili di stato e di controllo. Supponendo di introdurre vincoli sulle sole variabili di controllo:

$$\mathbf{u}_m \leq \mathbf{u}(k+i) \leq \mathbf{u}_M, \quad i = 0, ..., N-1,$$

dove la disuguaglianza tra i vettori va intesa elemento per elemento e i vettori $\mathbf{u}_m$ $\mathbf{u}_M$ contengono i valori minimi e massimi assegnabili alle variabili di controllo:

$$\mathbf{u}_m = \begin{bmatrix} u_{m,1} \\ u_{m,2} \\ \vdots \\ u_{M,m} \end{bmatrix},\ \mathbf{u}_M = \begin{bmatrix} u_{M,1} \\ u_{M,2} \\ \vdots \\ u_{M,m} \end{bmatrix}.$$

Si ottiene in questo modo il problema di ottimizzazione:

$$\min_{U(k)} \bar{H}(\mathbf{x}(k), U(k), k) = (\mathcal{A}\mathbf{x}(k)+\mathcal{B}U(k))^T\mathcal{Q}(\mathcal{A}\mathbf{x}(k)+\mathcal{B}U(k))+U(k)^T\mathcal{R}U(k)$$

con i vincoli:

$$\mathcal{X}(k) = \mathcal{A}\mathbf{x}(k) + \mathcal{B}U(k)$$
$$\mathbf{u}_m \leq U(k) \leq \mathbf{u}_M \,.$$

Questo problema non ammette più una soluzione esplicita (come in 3.5), ma può essere risolto con metodi di programmazione quadratica. In conclusione, l'approccio in anello aperto equivale ad un problema di ottimizzazione in cui è possibile includere esplicitamente vincoli su variabili di controllo e stati futuri.

3.3.2 Receding Horizon

Il problema di controllo ottimo su tempo finito porta alla determinazione di una legge di controllo tempo variante definita sull'orizzonte di predizione. Per ricondursi ad una legge di controllo tempo invariante e definita ad ogni istante k, è possibile applicare l'approccio detto *Receding Horizon* (o a orizzonte mobile). Ad ogni istante k viene risolto il problema di ottimizzazione su orizzonte finito $[k, k+N]$ e si applica solo il primo ingresso $\mathbf{u}^o(k)$ della matrice $U^o(k)$ calcolata. Al successivo istante $k+1$ si ripete l'ottimizzazione con riferimento all'orizzonte di predizione $[k+1, k+N+1]$. Il principio RH fornisce dunque una legge di controllo invariante nel tempo e basata sempre sullo stato corrente:

$$\mathbf{u} = K_{RH}(x)$$

Nei problemi soggetti a vincoli questa è una legge in retroazione di tipo implicito sullo stato, mentre nel caso di problemi senza vincoli questa coincide con il primo elemento della soluzione in anello aperto:

$$\mathbf{u}(k) = -K(0)\mathbf{x}(k)$$

Nel caso di problemi con vincoli non è possibile calcolare la soluzione dell'equazione di Riccati con le tecniche introdotte, ma essa rapresenta comunque la soluzione a cui idealmente tendere.

3.3.3 Inseguimento di riferimenti

Si suppone che il sistema sia descritto da:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)\,,$$

in cui $\mathbf{x} \in \mathbb{R}^n$ è misurabile. L'obiettivo del problema è seguire un riferimento $\mathbf{y}^o$ nell'orizzonte di predizione. Definendo:

$$\mathcal{Y}^o(k) = \begin{bmatrix} \mathbf{y}^o(k+1) \\ \mathbf{y}^o(k+2) \\ \vdots \\ \mathbf{y}^o(k+N-1) \\ \mathbf{y}^o(k+N) \end{bmatrix},$$

Si dimostra che è possibile ottenere:

$$\bar{H}(\mathbf{x}(k), U(k); k) = (\mathcal{Y}^o(k) - \mathcal{X}(k))^T \mathcal{Q}(\mathcal{Y}^o(k) - \mathcal{X}(k)) + U(k)^T \mathcal{R} U(k)\,,$$

che porta, nel caso non vincolato, alla soluzione:

$$U^o(k) = (\mathcal{B}^T \mathcal{Q} \mathcal{B} + \mathcal{R})^{-1} \mathcal{B}^T \mathcal{Q}(\mathcal{Y}^o(k) - \mathcal{A}\mathbf{x}(k))\,.$$

Nel caso si considerino vincoli sullo stato o sulle variabili di controllo sull'orizzonte di predizione, la soluzione deve essere calcolata numericamente.

3.3.4 Stabilità

In questa sezione si espone come derivare una legge di controllo che garantisca proprietà di stabilità per il sistema retroazionato. Si considera per semplicità il solo problema di regolazione a zero dello stato, supponendo che questo sia accessibile. Si assuma che il sistema sia descritto da:

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))\,,$$

in cui $f(\mathbf{x}(k), \mathbf{u}(k))$ è derivabile con continuità rispetto ai suoi argomenti. Si supponga inoltre che esista la legge di controllo ausiliaria:

$$\mathbf{u} = K_a(x)\,,$$

e un insieme invariante X_f contenente l'origine, tali che, partendo da uno stato iniziale $x(\bar{k})$ all'interno di X_f e applicando la legge di controllo ausiliaria, si rimane in X_f. Si consideri ora il problema di determinare la sequenza di variabili di controllo:

$$U(k) = [\mathbf{u}(k)^T, \mathbf{u}(k+1)^T, ..., \mathbf{u}(k+N-1)^T]^T\,,$$

che minimizzi la cifra di merito:

$$H(\mathbf{x}(k), U(k), k) \;=\; \sum_{i=0}^{N-1} (||\mathbf{x}(k+i)||_Q^2 + ||\mathbf{u}(k+i)||_R^2) + V_f(\mathbf{x}(k+N))\,,$$

con i vincoli:

$$\begin{aligned} &\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k))\,, \\ &\mathbf{x}(k+N) \in X_f \end{aligned}$$

V_f è un opportuno peso sullo stato alla fine dell'orizzonte di predizione. La soluzione del problema di ottimizzazione e l'applicazione del RH portano alla definizione della legge di controllo:

$$\mathbf{u} = K_{RH}(\mathbf{x})\,.$$

Nelle ipotesi esposte è possibile dimostrare che se per ogni $x \in X_f$ è soddisfatta la condizione:

$$V_f(f(\mathbf{x}(k)), K_a(\mathbf{x}(k))) - V_f(\mathbf{x}(k)) + (||\mathbf{x}(k)||_Q^2 + ||K_a(\mathbf{x}(k))||_R^2) \leq 0\,, \quad (3.7)$$

allora l'origine $x = 0$ è un punto di equilibrio asintoticamente stabile del sistema in anello chiuso con la legge di controllo RH. Supponendo ora che

il sistema sia descritto da:

$$\mathbf{x}(k+1) = A\mathbf{x}(k) + B\mathbf{u}(k)\,.$$

Utilizzando le stesse matrici di peso Q e R del problema di partenza, si determini il controllo LQ su orizzonte infinito con legge di controllo:

$$\mathbf{u} = -K_{LQ}\mathbf{x}(k)\,, \tag{3.8}$$

e la matrice P soluzione dell'equazione di Riccati stazionaria:

$$(A - BK_{LQ})^T P(A - BK_{LQ}) - P = -(Q + K_{LQ}^T BK_{LQ})\,.$$

Si consideri come legge di controllo la 3.8, come vincolo finale:

$$X_f = \{\mathbf{x} | \mathbf{x}^T P\mathbf{x} \leq \alpha\}\,,$$

dove α è uno scalare sufficientemente piccolo, e come peso finale:

$$V_f(\mathbf{x}) = \mathbf{x}^T P\mathbf{x}\,.$$

Con queste scelte è soddisfatta la eq. 3.7 e quindi la stabilità asintotica del sistema nell'origine [45].

3.3.5 Formulazione non lineare

L'approccio MPC lineare porta dunque alla formulazione di un problema di minimizzazione vincolata che presenta:

- una funzione di costo quadratica rispetto alle variabili di stato e di controllo;
- vincoli lineari rispetto a variabili di stato e di controllo.

Per questo tipo di problemi esistono numerosi solutori numerici che permettono di ottimizzare efficientemente la funzione di costo. Esistono numerose situazioni nelle quali è necessario introdurre delle non linearità nel problema, introducendo ad esempio funzioni di costo non quadratiche oppure vincoli non lineari. In questi casi viene definito il problema MPC non lineare (o NMPC), che risulta essere più complesso da risolvere dal punto di vista numerico. Considerando il generico sistema dinamico non lineare:

$$\mathbf{x}(k+1) \;=\; f(\mathbf{x}(k), \mathbf{u}(k))\,, \tag{3.9}$$

anche in questo caso è possibile formulare il problema di inseguimento di un riferimento e trovare la sequenza di variabili di controllo:

$$U(k) = [\mathbf{u}(k)^T, \mathbf{u}(k+1)^T, ..., \mathbf{u}(k+N-1)^T]^T\,,$$

che minimizza la cifra di merito su orizzonte finito:

$$H(\mathbf{x}(k), U(k), k) = \sum_{i=0}^{N-1} (||h_x(\mathbf{x}(k+i)) - \mathbf{y}_x(k+i)||^2_{W_x} + \\ +\; ||h_u(\mathbf{u}(k+i)) - \mathbf{y}_u(k+i)||^2_{W_u}) + ||h_N(\mathbf{x}(k+N)) - \mathbf{y}_x(N)||^2_{W_N}\,,$$

dove $\mathbf{x} \in \mathbb{R}^n$ è lo stato del sistema, $\mathbf{u} \in \mathbb{R}^m$ è il vettore di variabili di controllo e $\mathbf{y}_x$, $\mathbf{y}_u$ sono i riferimenti per stato e controllo, mentre $h_x(\cdot) : \mathbb{R}^n \to \mathbb{R}^n$ e $h_u(\cdot) : \mathbb{R}^m \to \mathbb{R}^m$ sono generiche funzioni di stato e variabili di controllo. La minimizzazione è soggetta ai vincoli della

dinamica del sistema (Eq. 3.9) ed eventualmente a vincoli aggiuntivi:

$$\begin{aligned}
\mathbf{x}^m_{k+i} &\leq \mathbf{x}(k+i) \leq \mathbf{x}^M_{k+i}\,, \forall i = 0, ..., N-1 \\
\mathbf{u}^m_{k+i} &\leq \mathbf{u}(k+i) \leq \mathbf{u}^M_{k+i}\,, \forall i = 0, ..., N-1 \\
\mathbf{c}^m_{k+i} &\leq c_{k+i}(\mathbf{x}(k+i), \mathbf{u}(k+i)) \leq \mathbf{c}^M_{k+i}\,, \forall i = 0, ..., N-1\,.
\end{aligned}$$

Dove si trovano i vincoli di disuguaglianza lineari, e il generico vincolo non lineare su stato e controllo definito sull'orizzonte di predizione tramite la funzione $c(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^c_{k+i}$. Come nel caso di MPC lineare, si applica l'approccio del *Receding Horizon* effettuando dunque una minimizzazione del funzionale di costo ad ogni passo.

In generale l'evoluzione dello stato sull'orizzonte di predizione differirà dalla reale evoluzione del sistema in anello chiuso. In primo luogo questo porta ad un'evoluzione dello stato che può essere significativamente diversa da quella ottima su tempo infinito. In secondo luogo, come per il caso lineare, l'applicazione del controllo predittivo non garantisce che il sistema in anello chiuso sia stabile, sebbene sia possibile dimostrare localmente l'asintotica stabilità del sistema linearizzato [45]. Il generico schema di controllo è rappresentato in fig. 3.2.

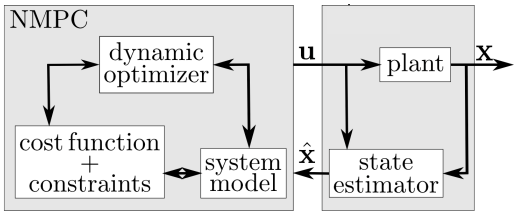


Figura 3.2: Schema di controllo NMPC

3.4 Applicazione del Controllo Predittivo al sistema

Uno degli obiettivi fondamentale del controllo di un manipolatore aereo può essere l'inseguimento di traiettorie o *set-point* da parte dell'*end-effector* del braccio robotico. Si tratta di un obiettivo difficile da raggiungere a causa della complessità del sistema complessivo (composto da più sottosistemi) e dalle approssimazioni introdotte descritte precedentemente (sez. 2.6). L'applicazione del Controllo predittivo al sistema in esame, in relazione all'obiettivo, richiede: la definizione del modello da utilizzare per effettuare la predizione, la definizione del funzionale di costo e l'introduzione di eventuali vincoli che devono essere rispettati. Sono stati studiati tre diversi modelli di predizione del sistema:

- modello interamente dinamico;
- modello in parte dinamico ed in parte cinematico;
- modello interamente cinematico.

Si espongono nelle seguenti sezioni i diversi modelli studiati, la definizione dei funzionali di costo, gli eventuali vincoli ed infine l'introduzione alla minimizzazione gerarchica.

3.4.1 Modello interamente dinamico

Nel caso di modello interamente dinamico per la predizione vegono usate le equazioni che descrivono la dinamica del sistema complessivo (2.6.3). In particolare il modello considera le seguenti variabli di stato e di controllo:

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} {}^{i}\mathbf{p}_b & {}^{i}\boldsymbol{\phi}_b & \mathbf{q} & {}^{i}\mathbf{v}_b & {}^{i}\dot{\boldsymbol{\phi}}_b & \dot{\mathbf{q}} \end{bmatrix}^T \\ \mathbf{u} &= \begin{bmatrix} \mathbf{0}_{6+n} & {}^{i}\mathbf{R}_b\,\mathbf{T} & \boldsymbol{\tau} & \boldsymbol{\tau}_q \end{bmatrix}^T , \end{aligned} \tag{3.10}$$

dove ${}^{i}\mathbf{p}_b$ e ${}^{i}\boldsymbol{\phi}_b$ sono rispettivamente vettore posizione e orientamento del drone in E_i (sistema di riferimento inerziale), ${}^{i}\mathbf{v}_b$ è il vettore velocità

traslazionale del drone in E_i, ${}^i\dot{\boldsymbol{\phi}}_b$ è la derivata temporale dell'orientamento del drone, mentre $\mathbf{q}$ e $\dot{\mathbf{q}}$ sono il vettore delle variabili di giunto del braccio robotico e la sua derivata temporale. Le variabili di controllo $\mathbf{T}$, $\boldsymbol{\tau}$ e $\boldsymbol{\tau}_q$ sono rispettivamente il vettore forza di *Thrust* nel sistema di riferimento E_b (sistema di riferimento solidale con il drone), le coppie sul telaio del drone, e le coppie ai giunti del braccio robotico. Si può dunque scrivere il sistema dinamico in forma matriciale:

$$\begin{bmatrix} \dot{\boldsymbol{\xi}} \\ \ddot{\boldsymbol{\xi}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbb{I}_\xi \\ \mathbf{0} & -I(\boldsymbol{\xi})^{-1}\, C(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}}) \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \dot{\boldsymbol{\xi}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ -I(\boldsymbol{\xi})^{-1}\, \mathbf{G}(\boldsymbol{\xi}) \end{bmatrix} + \mathbf{u} \ ,$$

con $\boldsymbol{\xi} = \begin{bmatrix} {}^i\mathbf{p}_e & {}^i\boldsymbol{\phi}_b & \mathbf{q} \end{bmatrix}^T$. $I(\boldsymbol{\xi})$ è la matrice d'inerzia del sistema complessivo, $C(\boldsymbol{\xi}, \dot{\boldsymbol{\xi}})$ rappresenta i termini centrifughi e di Coriolis, infine $\mathbf{G}(\boldsymbol{\xi})$ contiene gli effetti gravitazionali. Si nota che il sistema dinamico utilizzato per la predizione è non lineare.

3.4.2 Modello ibrido

Il secondo modello studiato considera la dinamica del sistema complessivo solo in parte. In particolare, viene considerata la dinamica del solo drone, poichè caratterizzato da una matrice di inerzia tempo invariante. In questo modo si evita di effettuare la valutazione della matrice d'inerzia del sistema complessivo ad ogni passo di predizione, risparmiando notevoli risorse computazionali. Per quanto riguarda la restante parte del sistema, viene considerata solo la cinematica, lasciando al controllo di basso livello il compito di seguire i riferimenti di velocità generati. In definitiva, il modello utilizzato coincide con quello presentato in 2.21, per quanto riguarda il drone. Per quanto riguarda il braccio può essere scritta la relazione cinematica:

$$\dot{\mathbf{q}} \ = \ \mathbf{u}_\mathbf{q} \, . \tag{3.11}$$

Dove $\mathbf{u}_\mathbf{q}$ è la variabile di controllo utilizzata dal controllo MPC per generare i riferimenti di velocità. Quindi il sistema complessivo può essere scritto

sotto forma matriciale:

$$\begin{bmatrix} {}^i\dot{\mathbf{p}}_b \\ {}^i\dot{\boldsymbol{\phi}}_b \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \mathbb{I}_{3\times3} & 0 & 0 \\ 0 & {}^i\boldsymbol{\Gamma}_b & 0 \\ 0 & 0 & \mathbb{I}_{4\times4} \end{bmatrix} \begin{bmatrix} {}^i\mathbf{v}_b \\ \boldsymbol{\omega} \\ \mathbf{u_q} \end{bmatrix} \tag{3.12}$$

$$\begin{bmatrix} {}^i\dot{\mathbf{v}}_b \\ \dot{\boldsymbol{\omega}} \end{bmatrix} = \begin{bmatrix} -\mathbf{g} \\ I_R^{-1}(-\boldsymbol{\omega} \times I_R\boldsymbol{\omega}) \end{bmatrix} + \begin{bmatrix} \mathbf{I}_T^{-1}\,{}^i\mathbf{R}_b & 0 \\ 0 & \mathbf{I}_R^{-1} \end{bmatrix} \begin{bmatrix} \mathbf{T} \\ \boldsymbol{\tau}_b \end{bmatrix}$$

dove $\mathbb{I}_{j\times j} \in \mathbb{R}^{j\times j}$ indica la matrice identità di opportune dimensioni. In definitiva le variabili di stato e di controllo utilizzate nel modello sono:

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} {}^i\mathbf{p}_b & {}^i\boldsymbol{\phi}_b & {}^i\mathbf{v}_b & {}^i\dot{\boldsymbol{\phi}}_b & \dot{\mathbf{q}} \end{bmatrix}^T \\ \mathbf{u} &= \begin{bmatrix} \mathbf{u_q} & \mathbf{T} & \boldsymbol{\tau} \end{bmatrix}^T, \end{aligned} \tag{3.13}$$

3.4.3 Modello interamente cinematico

L'ultimo modello studiato descrive solo la cinematica del sistema complessivo:

$$\begin{bmatrix} {}^i\dot{\mathbf{p}}_b \\ \dot{\phi} \\ \dot{\mathbf{q}} \end{bmatrix} = \begin{bmatrix} \mathbb{I}_{8\times8} \end{bmatrix} \begin{bmatrix} {}^i\mathbf{v}_b \\ \omega_\phi \\ \mathbf{u_q} \end{bmatrix}, \tag{3.14}$$

considerando il seguente vettore di variabili di stato e variabili di controllo:

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} {}^i\mathbf{p}_b & \phi & \mathbf{q} \end{bmatrix}^T \\ \mathbf{u} &= \begin{bmatrix} {}^i\mathbf{v}_b & \omega_\phi & \mathbf{u_q} \end{bmatrix}^T, \end{aligned} \tag{3.15}$$

dove le variabili di controllo relative al drone sono ${}^i\mathbf{v}_b = [\dot{x}\,\dot{y}\,\dot{z}] \in \mathbb{R}^3$ e $\omega_\psi \in \mathbb{R}$ ovvero le velocità traslazionali e di sola imbardata del drone. Si nota che passando dal modello dinamico al modello cinematico, si riducono i gradi di libertà del sistema. Infatti, considerando un veicolo sotto-attuato (dotato cioè di quattro attuatori a fronte di sei GdL), le variabili controllate per MPC sono solo quattro, in quanto gli angoli di rollio e beccheggio (ψ, ϑ) sono variabili di controllo per gli anelli di basso livello. Si nota infine che il sistema qui utilizzato è lineare.

3.4.4 Funzionale di costo

Come già introdotto precedentemente l'obiettivo del sistema è l'inseguimento di una traiettoria tramite l'*end-effector* del braccio robotico. Il funzionale di costo legato a questo obiettivo conterrà quindi l'errore di inseguimento dei riferimenti, ovvero la differenza tra la posizione desiderata e attuale dell'*end-effector*:

$$H_I = \sum_{i=0}^{N-1} (||{}^i\mathbf{p}_e(\mathbf{x}(i)) - \mathbf{y}^o_{ee}(i)||^2_{W_x} + ||\mathbf{u}(i)||^2_{W_u}) \,, \tag{3.16}$$

dove: ${}^i\mathbf{p}_e(\mathbf{x}(i))$ è la posizione attuale dell'*end-effector* dipendente dalle variabili di stato del sistema tramite le funzioni di cinematica non lineare introdotte in sez. 2.5, $\mathbf{y}^o_{ee}(i)$ è invece la posizione desiderata, in generale variante ad ogni step. Infine $||\mathbf{u}(i)||^2_{W_u}$ è il termine di moderazione del controllo.

Il funzionale di costo è definito nello stesso modo per tutti e tre i modelli utilizzati, ma ovvimente utilizzerà le variabili di controllo relative a ciascun modello come definite nella sezione precedente.

Avendo definito un funzionale di costo non lineare, il controllo MPC che si applica al sistema risulta essere di tipo non lineare. Questo anche nel caso di utilizzo del modello cinematico lineare per la predizione.

3.4.5 Vincoli

Per ciascun modello sono stati imposti degli opportuni vincoli alla minimizzazione per tenere conto di alcuni aspetti secondari:

- evitare la collisione tra manipolatore robotico e drone,
- vincoli sulle variabili di controllo.
- vincoli sull'orientamento del drone.

Collisioni tra manipolatore e drone

Nel controllo del sistema è necessario tenere conto dei possibili urti tra manipolatore e velivolo. Una condizione che garantisce di evitare queste

collisioni è che la distanza relativa tra i giunti che possono andare incontro a urti e la base mobile sia superiore ad un valore minimo (fig. 3.3). Ciò può essere scritto in termini di posizione dei giunti del braccio robotico nella terna E_b. Facendo riferimento a sez. 2.5 è possibile ricavare la posizione di tali giunti in funzione delle variabili di stato e imporre:

$$\begin{aligned} {}^b z_3 &\geq 0.1\,[m] \\ {}^b z_e &\geq 0.1\,[m] \end{aligned} \tag{3.17}$$

dove ${}^b z_3$ e ${}^b z_e$ sono le distanze del terzo giunto e dell'*end-effector* dal piano xy della piattaforma mobile. Si nota che questi vaori sono contenuti nei vettori posizione del terzo giunto e dell'*end-effector* espressi nella terna E_b del drone (${}^b\mathbf{p}_3$ e ${}^b\mathbf{p}_e$ in sez. 2.5).

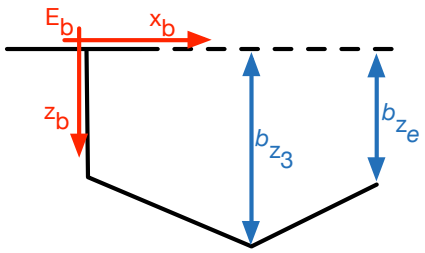


Figura 3.3: Rappresentazione del vincolo per evitare la collisione.

Vincoli sulle variabili di controllo

Per ciascun modello sono stati definiti dei vincoli sulle variabili di controllo per tener conto della saturazione degli attuatori.

- nei modelli che descrivono la dinamica del sistema sono stati imposti vincoli sulle variabili di controllo $\boldsymbol{\tau}$, $\boldsymbol{\tau}_m$, T:

$$\begin{aligned} T &\leq 40\,[N] \\ -1\,[Nm] \leq \boldsymbol{\tau} &\leq 1\,[Nm] \\ -1.5\,[Nm] \leq \boldsymbol{\tau}_m &\leq 1.5\,[Nm] \end{aligned}$$

Per le variabili di controllo del drone, ovvero forza di *Thrust* T e coppie sul telaio del drone $\boldsymbol{\tau}$, questi vincoli sono stati imposti dopo le misure delle forze e delle coppie generabili dal sistema (cap. 5). Per quanto riguarda le coppie ai giunti, ovvero $\boldsymbol{\tau}_m$, si è imposto il valore massimo di coppia motore ricavato da scheda tecnica.

- nei modelli che considerano la sola cinematica del sistema, sono stati invece imposti vincoli sulle variabili ${}^{i}\mathbf{v}_b$, ω_ϕ, $\mathbf{u_q}$ considerando le velocità massime raggiungibili da drone e variabili di giunto durante un passo di predizione.

Vincoli sull'orientamento del drone

Nei modelli considerano la dinamica del sistema si è imposto che gli angoli angoli di rollio e beccheggio (ψ e ϑ) non superino i valori:

$$\begin{matrix} \psi < \pi/4\,[\text{rad}] \\ \vartheta < \pi/4\,[\text{rad}] \end{matrix} . \tag{3.18}$$

In questo modo si impedisce al drone di portarsi in condizioni lontane dalla condizione di stabilità (ovvero con gli stessi angoli nulli).

3.4.6 Introduzione della gerarchia

L'uso delle generiche funzioni che esprimono vincoli non lineari sul sistema permette l'introduzione di una gerarchia di obiettivi da raggiungere, ciascuno caratterizzato da un funzionale di costo. L'idea è di effettuare una prima minimizzazione del funzionale relativo all'obiettivo principale, calcolando in questo modo i valori minimi che tale funzione di costo può assumere. In seguito, viene effettuata l'ottimizzazione sul funzionale di costo relativo all'obiettivo secondario, imponendo che la funzione di costo, relativa al primo obiettivo, non cambi. Il processo può essere così riassunto:

- Ottimizzazione del primo funzionale: tale ottimizzazione avviene senza vincoli legati alla presenza di gerarchie nel sistema, ma risulta essere la classica risoluzione di un problema NMPC. Questa ci fornisce i valori del funzionale che non dovranno cambiare nell'ottimizzazione del secondo funzionale.

- Ottimizzazione del secondo funzionale: si impone come vincolo che il funzionale relativo al primo obiettivo non cambi e si effettua la minimizzazione sul funzionale legato all'obiettivo secondario. Il vincolo a cui il solutore è sottoposto è dunque:

  $$c_{k+i}(\mathbf{x}(k+i)) = 0, \quad \forall i = 0, ..., N-1$$

  dove $c_{k+i}(\mathbf{x}(k+i))$ è appunto il vincolo di uguaglianza che impone al funzionale relativo al primo obiettivo di non variare durante la seconda ottimizzazione.

Si noti come il procedimento permetta di implementare il numero di funzionali pari al numero di obiettivi desiderati. Per fare ciò è necessario aggiungere un'ottimizzazione vincolata per ognuno di essi, con il conseguente incremento dei costi computazionali.

Un obiettivo secondario che può essere raggiunto dal sistema in esame è l'allineamento dei baricentri di braccio robotico e drone, questo infatti permette una più agevole stabilizzazione orizzontale del drone, in quanto evita

lo sbilanciamento dato dalla forza peso. Tale obiettivo viene caratterizzato dal funzionale di costo:

$$H_{II} = \sum_{i=0}^{N-1} (||{}^{b}\mathbf{p}_{G,xy}(\mathbf{x}(i))||^2_{W_x} + ||\mathbf{u}(i)||^2_{W_u}) \tag{3.19}$$

che contiene ${}^{b}\mathbf{p}_{G,xy}(\mathbf{x}(i))$, vettore descrivente la posizione del baricentro del braccio robotico rispetto alla terna E_b (eq. 2.13), proiettato sul piano xy della terna stessa:

$${}^{b}\mathbf{p}_{G,xy}(\mathbf{x}(i)) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} {}^{b}\mathbf{p}_{G}(\mathbf{x}(i)) \,. \tag{3.20}$$

L'obiettivo è infatti l'allineamento dei baricentri lungo l'asse z della terna E_b, si suppone infatti che il drone non si allontani significativamente dalla condizione di angoli di rollio e beccheggio nulli.

3.5 Controllo di basso livello

In questa sezione presentiamo l'architettura del controllo di basso livello implementata a seconda del modello utilizzato all'interno del controllo di alto livello. La sezione si suddivide essenzialmente in tre parti: la prima e la seconda relativa al controllo del drone (controllo di assetto e di posizione), la terza relativa al controllo delle variabili di giunto del braccio. Le tarature dei controlli di basso livello utilizzati sono riportate in appendice A.

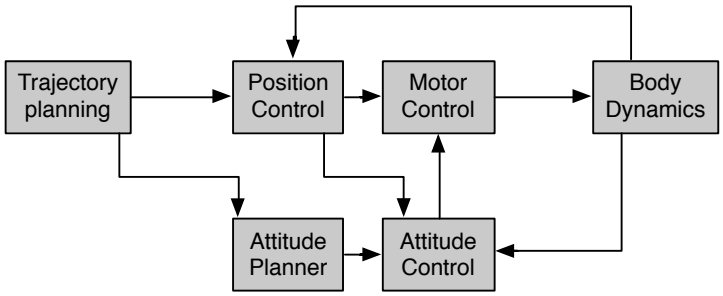


Figura 3.4: Interazione dei sistemi di controllo di alto e basso livello per il controllo del drone.

3.5.1 Controllo di assetto

A livello immediatamente superiore al controllo dei motori troviamo l'anello di stabilizzazione angolare del drone, ovvero il sistema di controllo che permette di controllare il valore degli angoli ϕ, ϑ e ψ. Questo anello è composto da una parte di controllo sulla velocità angolare (regolatore proporzionale-integrale) che prende come riferimento la variabile manipolata dall'anello di controllo di posizione angolare (regolatore proporzionale). Partendo dall'eq. 2.21, trascurando gli elementi extra diagonali della matrice di inerzia e linearizzado il sistema intorno alla condizione di *hovering*, si giunge all'equazione per il generico grado di libertà rotazionale:

$$\ddot{\psi} = \tau_\psi / J_\psi \,, \tag{3.21}$$

sulla quale viene tarato il nostro controllo di assetto, considerando come variabile di controllo la coppia τ_ψ.

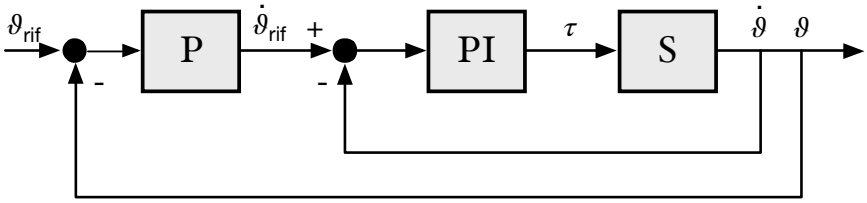


Figura 3.5: Anelli di Attitude Control. Dopo i blocchi dei regolatori troviamo il blocco rappresentante il sistema (nel caso di un solo grado di libertà contenente il solo effetto dell'inerizia.)

La taratura del sistema è stata effettuata imponendo una banda passante il più veloce possibile tenendo conto dei limiti sulle frequenze di campionamento delle misure. Nel corso delle simulazioni si mostrerà che le approssimazioni introdotte non pregiudicano le performance del sistema di controllo. Gli anelli di controllo sono raffigurati in fig. 3.5

3.5.2 Controllo di posizione

A livello immediatamente inferiore rispetto al controllo di alto livello troviamo il controllo di posizione, il quale riceve i riferimenti di posizione generati dal MPC. Questo sistema è stato progettato analogamente al controllo di assetto, realizzando un cosidetto P/PI e utilizzando come variabile di controllo direttamente la forza $\mathbf{T}$ generata dal drone in E_i (Eq. 2.21). Per quanto riguarda la dinamica del generico grado di libertà traslazionale, si ottiene:

$$\ddot{x} = T_x/m_x \tag{3.22}$$

Nel caso del grado di libertà z ovviamente l'equazione dovrà tenere conto anche del contributo gravitazionale che può essere compensato con un'opportuna azione in *feed-forward* del sistema di controllo:

$$\ddot{z} = T_z/m_z + g \tag{3.23}$$

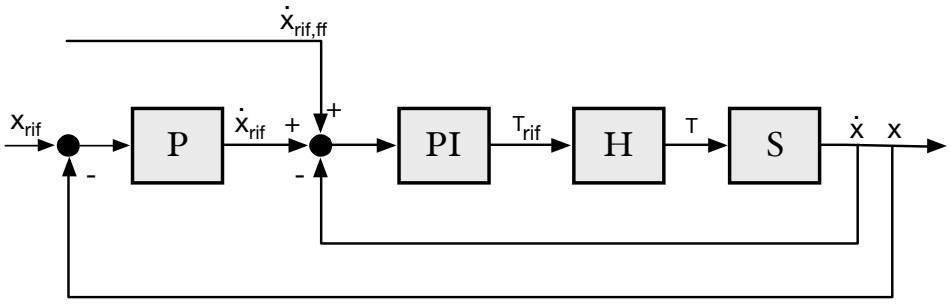


Figura 3.6: Anelli di controllo di posizione. Dopo i blocchi dei regolatori troviamo il blocco H rappresentante il controllo di assetto (approssimato come un filtro passa basso) che ci permette di seguire il riferimento di forza che utilizziamo come variabile di controllo nell'anello di posizione.

Come rappresentato in fig. 3.6 il sistema di controllo di posizione genera dei riferimenti per l'orientamento della forza di *Thrust* che vengono inviati al sistema di controllo di assetto. Questo si occupa di modificare i valori di rollio e beccheggio per inseguire la direzione della forza di controllo. Il legame tra forza di *Thrust* e angoli di rollio e beccheggio sono:

$$\begin{aligned} T_x &= |\mathbf{T}| \sin(\psi) \\ T_y &= |\mathbf{T}| \sin(\vartheta) \end{aligned} \quad . \tag{3.24}$$

Linearizzando attorno all'origine si ottiene:

$$\begin{aligned} T_x &\sim |\mathbf{T}| \psi \\ T_y &\sim |\mathbf{T}| \vartheta \end{aligned} \tag{3.25}$$

Si impongono di conseguenza i seguenti riferimenti al controllo di assetto:

$$\begin{aligned} \psi_{rif} &= T_x / |\mathbf{T}| \\ \vartheta_{rif} &= T_y / |\mathbf{T}| \end{aligned} \tag{3.26}$$

Si noti che la taratura dell'anello di posizione e velocità deve tenere conto delle prestazioni degli anelli di più basso livello (ovvero di assetto).

3.5.3 Controllo del braccio robotico

Per quanto riguarda infine il controllo del braccio robotico, in ciascuno dei servomotori del braccio si implementa lo stesso tipo di controllo P/PI. Ciò è possibile grazie all'elevato rapporto di trasmissione presente in ciascuno, si tratta infatti di trasmissioni con $n = 254$. Ciò ha reso possibile

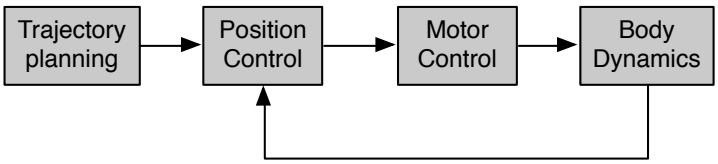


Figura 3.7: Sistema di controllo di posizionamento del braccio.

l'implementazione di un controllo decentralizzato, infatti la presenza di un rapporto di trasmissione elevato permette di trascurare il contributo d'inerzia lato *link* percepito a lato motore:

$$I_m^* = I_m + I_r/n^2 \tag{3.27}$$

con I_m inerzia del motore, I_r inerzia lato *link* e n rapporto di trasmissione. La taratura del controllo ha seguito la classica procedura per il regolatore P/PI. In fig. 3.7 si riporta lo schema di controllo di basso livello del braccio robotico e la sua interazione con il controllo di alto livello.

Capitolo 4

Simulazioni

4.1 Introduzione

In questa sezione si presentano le simulazioni che hanno seguito la sintesi del sistema di controllo. Le simulazioni sono state svolte in ambiente Matlab, appoggiandosi a due piattaforme open-source che hanno fornito il punto di partenza per la scrittura e la modifica del codice: *Robotics Toolbox* di Peter Croke [46], *Matlab NMPC routine* di Grüne e Pannek [47]. Partendo da questi *toolbox* è stato possibile implementare il modello del sistema complessivo e i diversi anelli di controllo. Dopo la prima validazione, simulazioni più realistiche sono state condotte sul software Gazebo, dove è stata testata l'architettura relmente funzionante sul sistema tramite simulazione *Software In The Loop (SITL)*. La sezione è così strutturata: nella prima parte vengono presentate le simulazioni effettutate in ambiente Matlab mostrando i risultati ottenuti con i tre modelli utilizzati sul controllore di alto livello, nella seconda parte viene invece presentata la struttura di controllo utilizzata nel *SITL* e i risultati ottenuti.

4.2 Simulazioni in ambiente Matlab

Nelle simulazioni condotte in ambiente Matlab è stato utilizzato un modello del sistema caratterizzato dagli stessi parametri dinamici stimati sperimentalmente sul sistema reale, che sono stati riportati in tab. 4.1. La

descrizione dettagliata delle equazioni dinamiche utilizzate per simulare il comportamento reale è riportata in sez. 2.6.

Tabella 4.1: Parametri dinamici utilizzati in ambiente Matlab.

Sistema	I_{xx} [kg m^2]	I_{yy} [kg m^2]	I_{zz} [kg m^2]	M [kg]
Drone	$4.29e-2$	$2.69e-2$	$7.21e-2$	3.485
Link 1	$6.976e-4$	$7.096e-4$	$3.911e-5$	$109e-3$
Link 2	$6.259e-5$	$6.547e-5$	$3.21e-5$	$39e-3$
Link 3	$1.531e-4$	$1.577e-4$	$3.278e-5$	$48e-3$

4.2.1 Modello Completamente Dinamico

La prima implementazione che è stata completata ha utilizzato un modello di predizione in grado di considerare gli effetti inerziali di tutto il sistema. In realtà le simulazioni che sono state portate a termine hanno evidenziato l'enorme complessità computazionale del controllo. Infatti, anche scegliendo orizzonti di predizione molto brevi, a causa della complessità del modello, il tempo richiesto dai passi di ottimizzazione era superiore alle 24 ore. Quindi, a causa dello scarso interesse pratico che tali simulazioni presentano, non vengono qui esposte.

4.2.2 Modello Ibrido

Per superare i problemi relativi alla complessità computazionale si è deciso di semplificare il modello di predizione. In questa prospettiva è stato scelto di costruire un modello che fosse per metà dinamico e per metà cinematico, ed utilizzarlo all'interno di MPC. Tramite questo controllore è necessario implementare un controllo di posizone nello spazio dei giunti solo per il braccio robotico. La simulazione condotta è stata effettuata con un intervallo di predizione composto da 10 step di durata 0.05 s l'uno, per un totale di 0.5 s di predizione nel futuro. Le matrici dei pesi utilizzati sono state:

- $Q = \text{diag}\left(\begin{bmatrix}20 & 20 & 20\end{bmatrix}\right)$ per l'errore di inseguimento di traiettoria

- pesi di 0.1 per le variabili di controllo legate al controllo del drone (coppie e forza di *Thrust*), pesi di 0.001 per le variabili legate al controllo del braccio robotico (velocità dei giunti).

$$R = 0.1 * \text{diag}\left(\begin{bmatrix} 1 & 1 & 1 & 1 & 0.01 & 0.01 & 0.01 & 0.01 \end{bmatrix}\right) \tag{4.1}$$

La simulazione ha come scopo l'inseguimento di una traiettoria da parte dell'*end-effector* e i risultati sono riportati in fig. 4.1 e fig. 4.2.

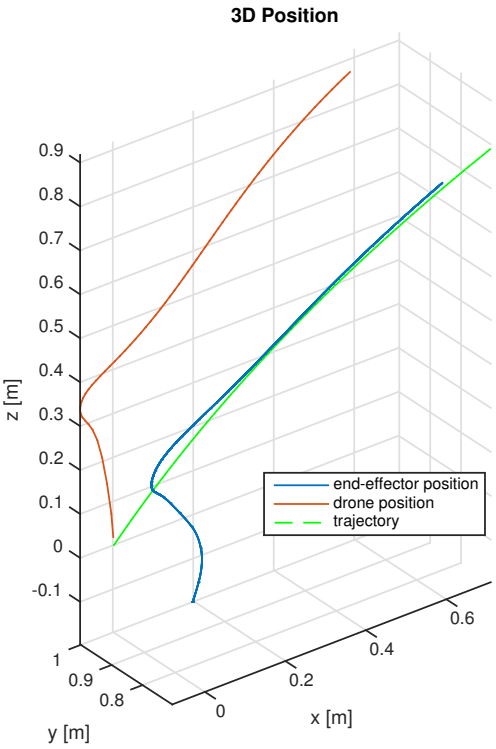


Figura 4.1: Rappresentazione 3D della traiettoria inseguita.

Come si può notare dalle immagini riportate il sistema sembra inseguire il riferimento di posizione inviato. La linea rossa rappresenta la traiettoria del drone, che partendo dalla posizione $\begin{bmatrix} 0, & 1, & 0 \end{bmatrix}$ si solleva e si porta in una posizione tale da permettere all'*end-effector*, la cui traiettoria è rappresentata in blu, di seguire il riferimento, riportato in verde. Si nota che pur

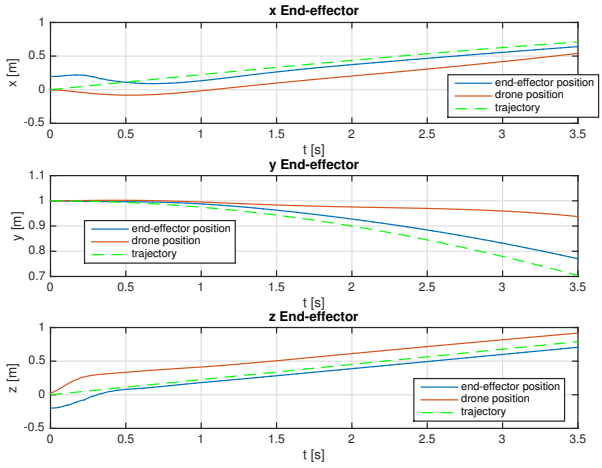


Figura 4.2: Rappresentazione della traiettoria inseguita dall'end effector.

essendo posizione iniziale del drone e riferimento iniziale sovrapposti, il controllo MPC genera una traiettoria coerente con le possibilità di movimento offerte dal braccio robotico. Le tempistiche di simulazione risultano tuttavia limitate dall'ancora elevato costo computazionale. Pur non essendo elevate come nel caso del controllore MPC totalmente dinamico, risultano però ancora del tutto inadeguate per un'implementazione in *real time* del controllore. Infatti i tempi richiesti dall'ottimizzatore per minimizzare il funzionale di costo, nell'orizzonte di predizione, sono caratterizzati da una media di $1164\,s$ (fig. 4.3). La taratura del controllore di basso livello del braccio robotico è riportata in appendice A.

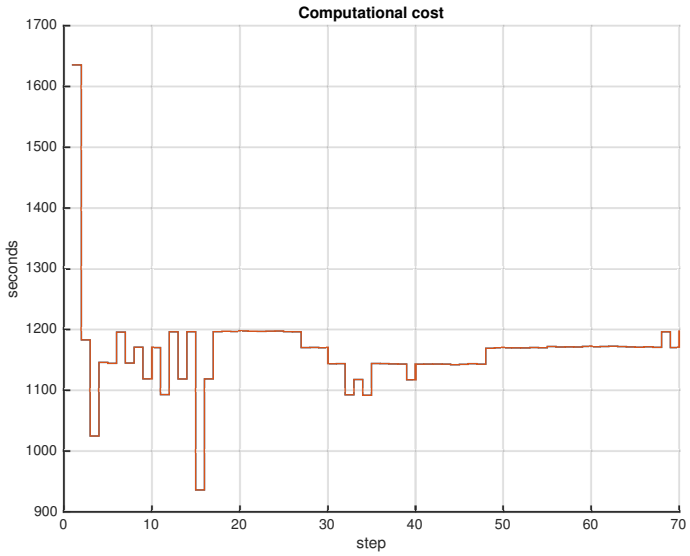


Figura 4.3: Tempi computazionali di simulazione. Durante ciascuno step vengono calcolati i valori ottimi delle variabili di controllo sui 10 step futuri.

4.2.3 Modello Cinematico

Presentiamo infine i risultati di simulazione ottenuti implementando sul controllore di alto livello il modello cinematico del sistema complessivo. In particolare, in questa sezione presentiamo e confrontiamo i risultati ottenuti dalle simulazioni con e senza gerarchia. Grazie alla presenza dei controllori di più basso livello che garantiscono il mantenimento della condizione di posizione di equilibrio stabile del sistema, il controllore di alto livello viene utilizzato per generare i riferimenti di velocità e posizione come esposto in sez. 3.4.3. L'invio dei riferimenti avviene ad una frequenza che permette al controllo di basso livello di raggiungere i riferimenti di velocità assegnati (la cui taratura del controllo di basso livello è riportata in Appendice A). Per questo motivo scegliamo passi di predizione di 2 s ed un orizzonte temporale di quattro passi, per un totale di 8 s nel futuro. Le matrici utilizzate nel funzionale di costo sono state:

- $Q = \text{diag}\left(\begin{bmatrix} 10^2 & 10^2 & 10^2 \end{bmatrix}\right)$ per l'errore di inseguimento di traiettoria
- pesi di 0.1 per le variabili di controllo legate al controllo del drone (velocità lineari $\dot{x}\,\dot{y}\,\dot{z}$ e velocità di imbardata ω_ψ), pesi di 0.001 per le variabili legate al controllo del braccio robotico (velocità dei giunti).

$$R = 0.1 * \text{diag}\left(\begin{bmatrix} 1 & 1 & 1 & 1 & 0.01 & 0.01 & 0.01 & 0.01 \end{bmatrix}\right) \tag{4.2}$$

Anche in questo caso la scelta di queste matrici è stata effettuata dopo numerose simulazioni volte ad ottenere un miglioramento del comportamento del sistema. La traiettoria inseguita dall'*end-effector* è composta da 3 punti di passaggio disposti alla distanza di 1 m l'uno dall'altro, nelle tre coordinate. I risultati sono riportati in fig. 4.5 e fig. 4.4, dove viene raffigurata la traiettoria e l'errore di posizionamento dell'*end-effector*. Si osserva innanzitutto che il sistema insegue il riferimento assegnato, portando il drone in prossimità della posizione desiderata e permette l'abbattimento dell'errore a transitorio esaurito. In secondo luogo si può osservare la presenza di sovraelongazioni nella risposta del sistema. Tali sovraelongazioni sono causate dai riferimenti di velocità, generati dal sistema di alto livello,

ed inviati in *feed-forward* al sistema di controllo di basso livello. Infine si può notare come le variabili di posizione dell'*end-effector* siano tra loro accoppiate, infatti il raggiungimento di un riferimento in una variabile genera dei disturbi nelle altre.

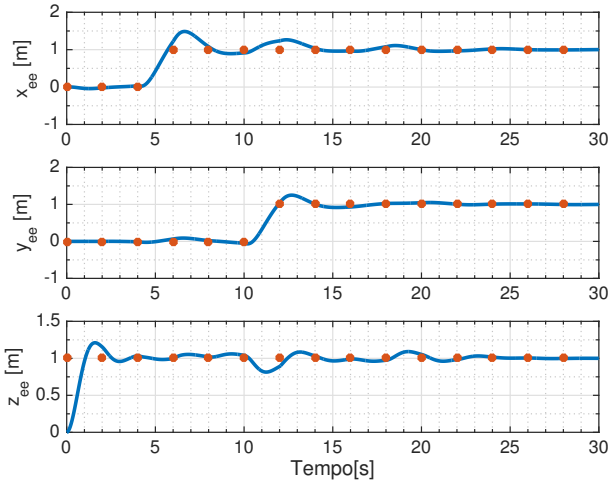


Figura 4.4: Traiettoria dell'*end-effector*.

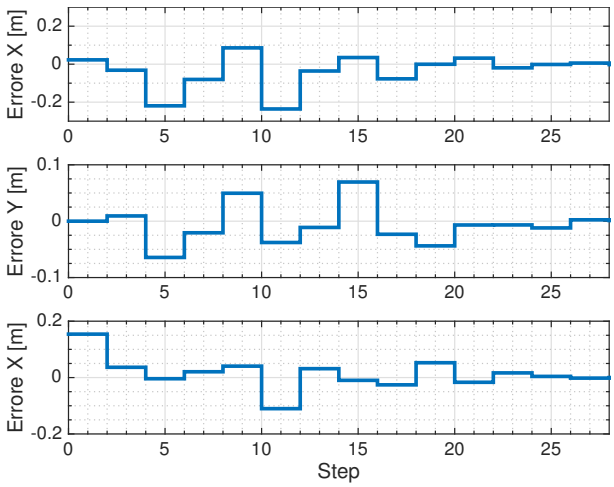


Figura 4.5: Errore di posizionamento dell'*end-effector*.

4.2.4 Modello Cinematico con Gerarchia

Nel controllo MPC, che utilizza il modello cinematico del sistema, è stato integrato l'approccio gerarchico tra i due funzionali di costo introdotti nelle sez. 3.4.4 e sez. 3.4.6. Per la taratura dei controllori di alto e basso livello sono stati utilizzati gli stessi parametri presentati per il modello senza gerarchia. Per quanto riguarda la funzione di costo addizionale di allineamento dei baricentri è stato scelta una matrice dei pesi:

$$Q = \text{diag}\left(\begin{bmatrix} 10^4 & 10^4 & 10^4 \end{bmatrix}\right) .$$

Tale matrice è stata tarata tramite prove simulative allo scopo di ottenere un buon comportamento del sistema. Sono stati ottenuti i risultati riportati in fig. 4.7 e fig. 4.6.

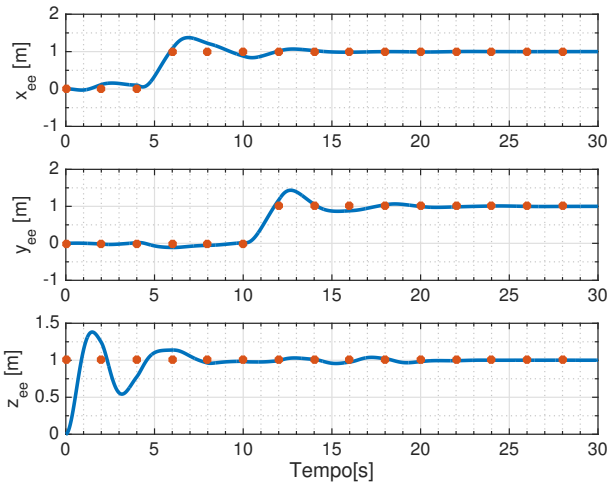


Figura 4.6: Traiettoria dell'*end-effector*.

Innanzitutto si nota anche in questo caso che il sistema insegue i riferimenti dati all'*end-effector*. In secondo luogo si nota una sovraelongazione sulla coordinata z probabilmete provocata da un movimento del braccio necessario per ridurre il disallineamento iniziale.

Si riportano in fig.4.8 i valori dei funzionali di costo che si ottengono implementando la gerarchia dei *task*. Si nota che il controllore con approccio gerarchico permette di ottimizzare il secondo funzionale di costo,

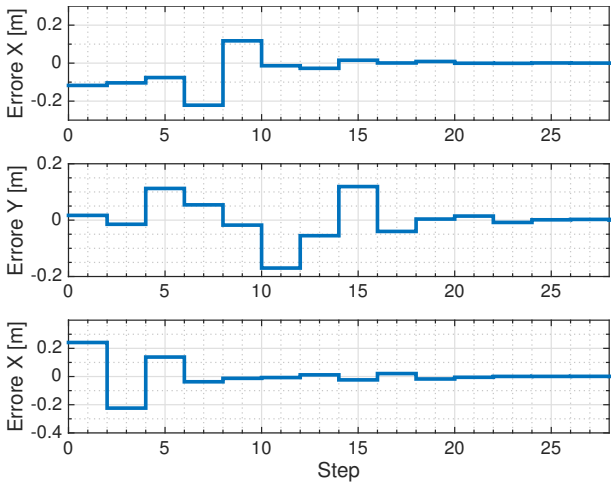


Figura 4.7: Errore di posizionamento dell'*end-effector*.

ottenendo un disallineamento dei baricentri inferiore, ma allo stesso tempo mantenendo un valore di errore dell'inseguimento della traiettoria confrontabile con quello ottenuto dal sistema con controllore non gerarchico. Si riportano le tempistiche ottenute tramite i controllori implementati. Dalla fig. 4.9 vediamo come i tempi computazionali siano in entrambi i casi simulati contenuti rispetto a quelli ottenuti dagli altri modelli. Ovviamente i tempi computazionali del controllore che considera la gerarchia risultano essere mediamente superiori a causa della doppia ottimizzazione richiesta. Si noti che il tempo computazionale riportato si riferisce a simulazioni che utilizzano un algoritmo di controllo non ottimizzato nè compilato. I tempi computazionali del setup sperimentale saranno notevolmente più contenuti.

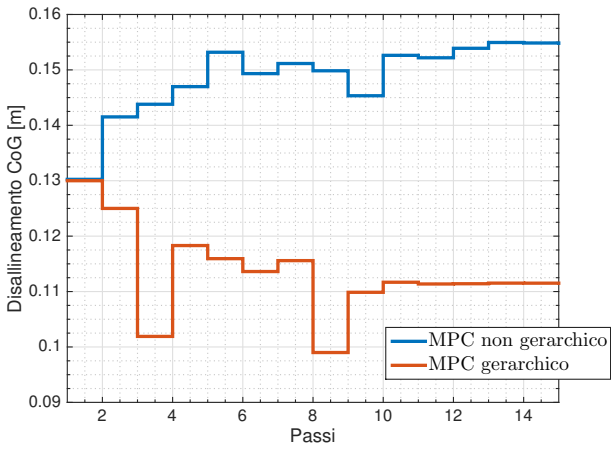


Figura 4.8: Disallineamento dei baricentri con e senza approccio gerarchico.

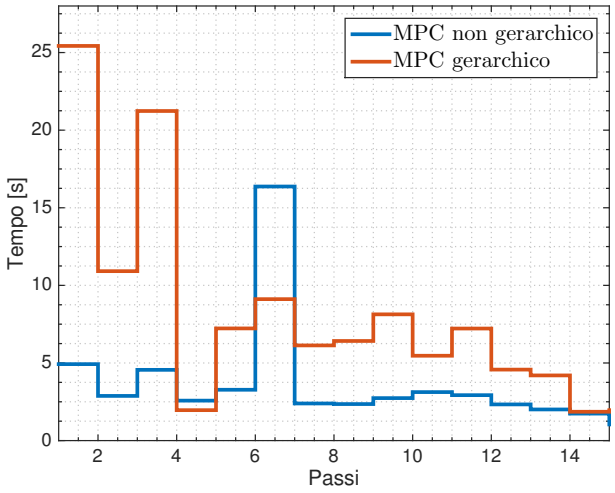


Figura 4.9: Tempo computazionale di ottimizzazione.

4.3 Simulazioni Software in The Loop

4.3.1 ROS - Robot Operating System

La sigla ROS (Robot Operating System) sta a indicare una struttura software *open-source* ampiamente utilizzata nell'ambito della robotica. La principale filosofia che sta alla base del progetto ROS è rendere semplice la comunicazione tra piattaforme diverse. In questo modo ROS incentiva la condivisione del codice online, facilitando l'integrazione dei sistemi.

ROS si sviluppa nel 2007 grazie allo Stanford Artificial Intelligence Laboratory (SAIL) e da allora ha subito una forte crescita che lo ha portato ad essere una delle piattaforme più utilizzate nell'ambito della robotica. Ciò è stato permesso dall'impegno di molti istituti di ricerca che hanno iniziato e sviluppato progetti in ROS pubblicandoli senza restrizioni online seguendo la filosofia dell'*open-source software*. Il codice sviluppato in ROS è facilmente verificabile in simulazione grazie ai simulatori 2D e 3D disponibili per la piattaforma. Un esempio di questi è Gazebo [48], un simulatore dinamico di robot in ambiente virtuale. Questo è in grado di simulare gruppi di robot, sensori e oggetti in un modello di ambientazione virtuale 3D. Può inoltre simulare il feedback dei sensori e interazioni fisiche realistiche tra gli oggetti.

Essenzialmente ROS si basa sulla costruzione di una struttura di processi chiamati *nodi*. Ognuno di questi assume funzioni specifiche e la rete di nodi che viene costruita è coordinata da un *nodo* principale detto *roscore*. I diversi *nodi* sono messi in comunicazione tramite *topic* messi a disposizione dal *roscore*. Ogni nodo ha la possibilità di iscriversi ai *topic* oppure pubblicare messaggi sui *topic* per scambiare informazioni. Un insieme di *nodi* viene solitamente raggruppato in un *package*, che contiene librerie, eseguibili e codice sorgente.

4.3.2 Simulatore Gazebo

Come già anticipato sono state condotte alcune simulazioni anche tramite il software ROS-Gazebo sfruttando i modelli e parte del software

messo a disposizione dal progetto Pixhawk [49], la piattaforma hardware utilizzata per il controllo del sistema. Nel dettaglio il sistema utilizzato per la simulazione è composto da più *nodi* ROS tra loro comunicanti che si occupano ciascuno di funzioni precise. L'architettura software complessiva per il controllo del drone risulta essere piuttosto complessa ed è rappresentata in fig. 4.11 nel caso della simulazione. I diversi nodi comunicano tra loro pubblicando e leggendo i messaggi scambiati attraverso dei *topic* ROS. Per comodità riportiamo solo la struttura formata dai nodi (i cerchi in fig. 4.11), mentre tralasciamo la rappresentazione i diversi *topic* dove i messaggi vengono letti e pubblicati. Entrando nel dettaglio il nodo *commander* viene utilizzato per scegliere la modalità di volo (manuale o automatica). L'utente può pubblicare messaggi attraverso il nodo *setpoint position* per definire un riferimento di posizione nello spazio cartesiano. I nodi *position control* e *attitude control* realizzano gli anelli di controllo di posizione e di assetto. Si passa infine ai nodi di controllo dei motori (anch'essi simulati). I messaggi riguardanti il controllo dei motori vengono infine spediti alla piattaforma di simuazione Gazebo che restituisce i valori delle variabili di stato del sistema. I nodi *attitude estimator* e *position estimator* non sono quelli utilizzati sul sistema reale in quanto non simuliamo la presenza di sensori virtuali, ma semplicemente aggiungiamo al segnale restituito dal simulatore del rumore bianco. Si osserva infine che i *nodi* hanno una frequenza di funzionamento di 200 Hz e sono tra loro sincronizzati tramite un topic *clock* che scandisce il tempo della simulazione. Raggruppando i nodi per realtà applicativa notiamo: in blu i nodi che verranno utilizzati realmente sulla scheda di controllo, in arancio il nodo di simulazione ed in nero nodi secondari di comunicazione.

I nodi ROS che si occupano della realizzazione degli anelli di controllo sono stati tarati sui parametri di inerzia caratteristici del sistema in esame, questi sono stati utilizzati anche per caratterizzare il modello del drone utilizzato in simulazione. Per quanto riguarda il modello utilizzato, si evidenzia che il motore di simulazione Gazebo permette l'utilizzo di modelli 3D descritti tramite file URDF (Universal Robot Description Format), ovvero un codice in linguaggio XML che permette di descrivere tutti gli elementi, le caratteristiche e i parametri di un Robot. Conoscendo i

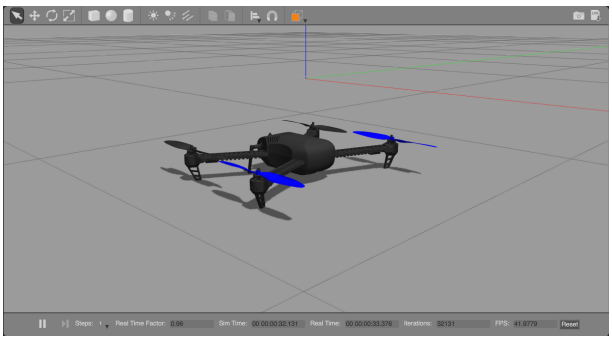


Figura 4.10: Modello 3D di drone Iris.

parametri dinamici, Gazebo è quindi in grado di simulare il comportamento dinamico del sistema. Si è scelto, in particolare, di utilizzare il modello 3D del drone *Iris*, perchè disponibile online liberamente come modello URDF. In simulazione è stato testato il comportamento del solo drone, senza la presenza del manipolatore, in quanto il modello URDF di un manipolatore simile a quello in esame non era disponibile. La taratura degli anelli di controllo ha seguito la classica procedura già seguita per le simulazione Matlab.

4.3.3 Simulazioni in ambiente Gazebo

L'implementazione del controllo MPC sul sistema complessivo è stata simulata tramite Gazebo creando un opportuno nodo ROS in grado di generare la traiettoria. Questo ha reso necessario l'utilizzo di un toolbox di ottimizzazione, che permettesse la risoluzione del problema di minimizzazione vincolata non lineare. Per ragioni di efficienza e versatilità ci siamo indirizzati sulla piattaforma ACADO [50].

ACADO Toolkit è un ambiente software contenente una collezione di algoritmi studiato per applicazioni di controllo automatico e ottimizzazione dinamica. Fornisce un'interfaccia per l'utilizzo di una grande varietà di algoritmi nelle più diverse applicazioni: controllo ottimo, stima dei parametri, controllo robusto e controllo predittivo. ACADO è inoltre scritto interamente in codice C++, si interfaccia facilmente con il software MATLAB ed è stato implementato come *ROS package* da *Clearpath Ro-*

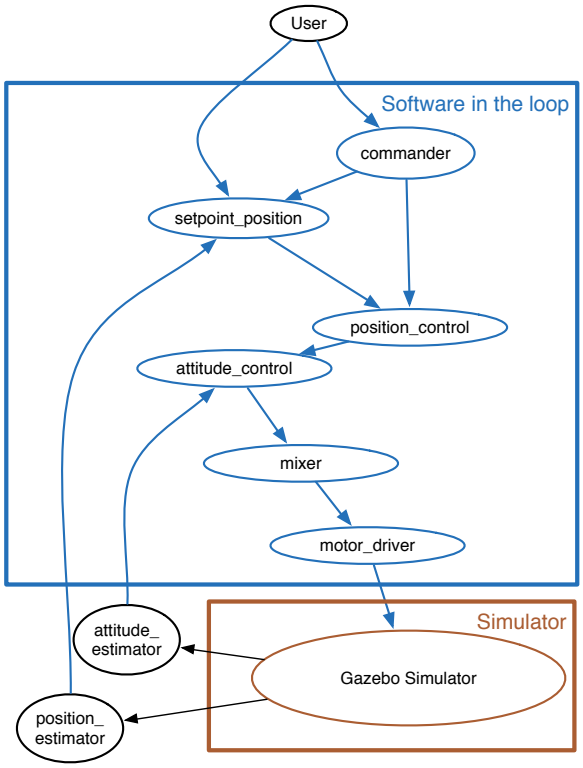


Figura 4.11: Architettura ROS in simulazione.

botics. L'implementazione è stata condotta realizzando un nodo ROS in grado di leggere i topic relativi allo stato del sistema, scrivere i riferimenti relativi allo stato del sistema, utilizzare le librerie di ACADO relative all'ottimizzazione e risolvere il probema di minimizzazione. Lo schema dei nodi di controllo e le loro connessioni utilizzate sono rappresentati in fig. 4.12, dove è evidenziato in verde il nodo ROS sviluppato e messo in comunicazione con gli altri nodi del sistema.

Il nodo che è stato sviluppato ha dunque lo scopo di leggere la posizione del robot, e calcolare posizione e velocità del drone da inviare come riferimento al controllo di basso livello per minimizzare una determinata funzione di costo. Il modello per la predizione utilizzato in ROS è quello che descrive la cinematica del solo quadricottero. Nel dettaglio il modello utilizzato per la predizione è lo stesso presentato in 3.4.3, ma senza

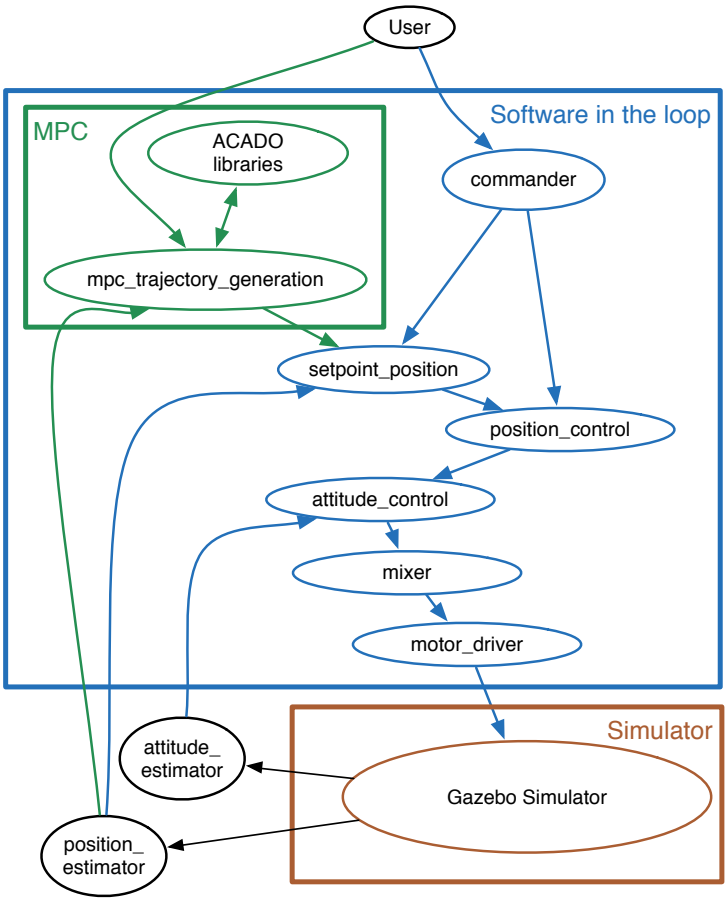


Figura 4.12: Architettura ROS in simulazione con generazione di traiettoria.

considerare il braccio:

$$^{i}\dot{\mathbf{p}}_b = {}^{i}\mathbf{v}_b \,,$$

dove $^{i}\mathbf{p}_b$ è la posizione del drone, mentre $^{i}\mathbf{v}_b$ è la velocità dello stesso. Il vettore di stato e degli ingressi sono dunque:

$$\mathbf{x} = {}^{i}\mathbf{p}_b$$
$$\mathbf{u} = {}^{i}\mathbf{v}_b \,,$$

mentre come funzione di costo da minimizzare si sceglie:

$$H(\mathbf{x}(k), \cdot\,, k) \;=\; \sum_{i=0}^{N-1} \left(||(\mathbf{y}^o(k+i) - \mathbf{x}(k+i))||_Q^2 + ||\mathbf{u}(k+i)||_R^2 \right) ,$$

dove $\mathbf{y}^o(k+i)$ sono i riferimenti di posizione che vogliamo raggiungere. Ad ogni passo viene minimizzata la funzione di costo $H(\mathbf{x}(k), \cdot, k)$ e vengono pubblicati i riferimenti di posizione e velocità ottenuti. Per quanto riguarda le matrici dei pesi, esse sono state scelte dopo alcuni tentativi di simulazione con l'obiettivo di ottenere buone prestazioni:

$$Q = \text{diag}\begin{bmatrix}1 & 1 & 1\end{bmatrix}$$
$$R = \text{diag}\begin{bmatrix}0.1 & 0.1 & 0.1\end{bmatrix}.$$

Sono stati inoltre imposti vincoli sulle velocità massime del drone di 1 m/s in direzione z e 0.3 m/s in direzione xy, per assicurarci di generare posizioni e velocità che possono essere raggiunte dal nostro sistema. Particolare importanza assume la frequenza di pubblicazione dei messaggi da parte del controllo di alto livello. É infatti necessario che i riferimenti pubblicati vengano aggiornati in modo che il sistema di controllo di basso livello sia in grado di raggiungere i riferimenti di velocità imposti.

Per quanto riguarda il controllore di basso livello si è implementato lo stesso presentato in sez. 3.5, ovvero un P/PI in cascata su velocità e posizioni traslazionali. Ne riportiamo sinteticamente le caratteristiche:

$$\omega_{c,\vartheta} = 25\,[\text{rad/s}]$$
$$\omega_{c,v} = K_{pv}/m_q = 5\,[\text{rad/s}]$$
$$\omega_z = \omega_{cv,p}/10 = K_{iv}/K_{pv} = 0.5\,[\text{rad/s}]$$
$$\omega_{c,p} = \omega_{cv,p}/10 = 0.5\,[\text{rad/s}]$$

dove ω_{c_ϑ} è la banda passante dell'anello di assetto, $\omega_{cv,p}$ è la banda passante dell'anello di velocità traslazionale, ω_z è la frequenza a cui poniamo lo zero del nostro controllore e $\omega_{cp,p}$ è la banda passante dell'anello di posizione.

Per quanto riguarda le simulazioni in ambiente Gazebo, si riporta il

grafico riporante il confronto tra la risposta al gradino del sistema in presenza o in assenza del controllo di alto livello. In particolare il grafico riporta l'inseguimento del gradino in direzione x.

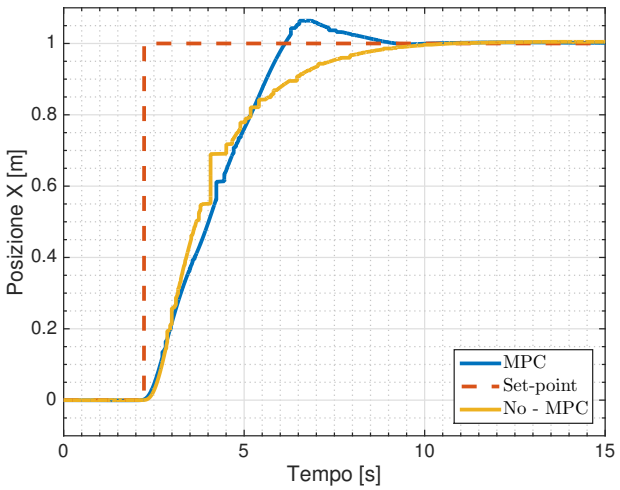


Figura 4.13: Risposta al gradino in presenza o assenza di generazione di traiettoria.

Si nota che la risposta del sistema con MPC è leggermente in anticipo, ma le risposte sono indubbiamente confrontabili. Si nota in particolare una piccola sovraelongazione nel caso sia presente il controllo di alto livello. Questo perchè vengono inviati riferimenti di velocità in *feed-forward* al sistema di controllo di basso livello.

Si riporta infine la risposta ad un riferimento costante nelle tre coordinate. I riferimenti costanti inviati sono ad una distanza di 3 m dal punto di partenza per le coordinate x e y, mentre di 4,2 m per la coordinata z, ovvero $\mathbf{y}^o = [5\ 5\ 8]$ (fig. 4.14 e fig. 4.15).In questa simulazione si osserva che il sistema di controllo di posizione di basso livello si porterebbe in condizione di instabilità prendendo in ingresso un riferimento di questo tipo poichè, per raggiungerlo, si allontanerebbe troppo dalla condizione di linearizzazione. Il controllo di alto livello ci permette invece di raggiungere efficaciemente il *set-point* inviato poichè genera posizioni intermedie e velocità intermedie di riferimento che possono essere raggiunte dal drone

senza portarlo in lontano dalla condizione di linearizzazione. Il riferimento è infine raggiunto con una piccola sovraelongazione sulle direzioni xy.

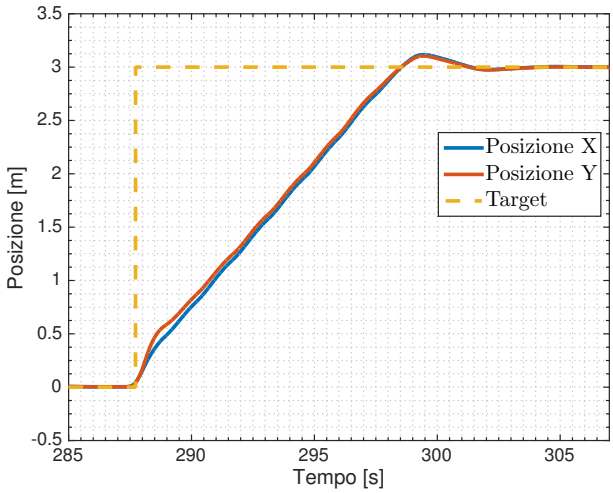


Figura 4.14: Risposta del sistema ad un riferimento costante sul piano xy

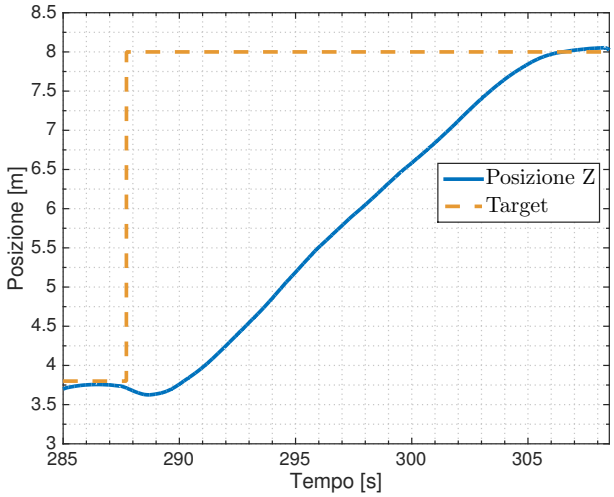


Figura 4.15: Risposta del sistema ad un riferimento costante in z

Capitolo 5

Prove Sperimentali

5.1 Introduzione

In questo capitolo si espongono i risultati ottenuti dalle prove sperimentali effettuate. Si presentano in particolare:

- Prove sperimentali per la stima della relazione funzionale tra variabili di controllo e forze effettivamente generate dalle eliche del drone.
- Prove sperimentali per la taratura del controllo di assetto.
- Prove sperimentiali di controllo di posizione del solo drone.
- Prove di controllo MPC per il solo braccio robotico..

5.2 Relazione tra variabili di controllo e forze aerodinamiche

L'uscita del controllo d'assetto è costituita dalle coppie desiderate che devono essere generate sul sistema. Calcolate queste coppie desiderate, è necessario conoscere le caratteristiche aerodinamiche e geometriche del drone, ed in particolare la relazione che lega le velocità di rotazione delle eliche con le coppie generate sulllo stesso (sez. 2.6.1). Nel drone qui

considerato la relazione geometrica è:

$$\begin{bmatrix} T \\ \tau_x \\ \tau_y \\ \tau_z \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ ds_\alpha & -ds_\alpha & -ds_\alpha & ds_\alpha & -ds_\alpha & ds_\alpha & ds_\alpha & -ds_\alpha \\ -dc_\alpha & -dc_\alpha & dc_\alpha & dc_\alpha & -dc_\alpha & -dc_\alpha & dc_\alpha & dc_\alpha \\ c & -c & c & -c & c & -c & c & -c \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ ... \\ T_8 \end{bmatrix}$$

che può essere riscritta:

$$\mathbf{T}_\tau = M\,\mathbf{T}_m \tag{5.1}$$

dove d è la distanza tra il motore e il centro del velivolo, s_α e c_α rappresentano rispettivamente il seno e il coseno dell'angolo tra l'asse di simmetria frontale e i bracci anteriori e c è un coefficiente che rappresenta il rapporto tra i coefficienti di thrust e drag delle eliche. In teoria, note le coppie desiderate sul drone sarebbe necessario invertire la matrice M (detta *Mixer*) per ricavare le forze che devono essere generate da ciascuna elica. Note le forze, si possono calcolare le velocità di rotazione. L'inversione della relazione può essere realizzata utilizzando la pseudoinversa della matrice M, poichè si ha di fronte una matrice rettangolare che da luogo ad un sistema sotto-dimensionato. Noto il vettore $\mathbf{T}_\tau$ esistono infatti infiniti vettori $\mathbf{T}_m$ che soddisfano la relazione. La relazione può essere dunque invertita utilizzando la matrice:

$$M^\dagger = M^T(MM^T)^{-1}\,. \tag{5.2}$$

É opportuno osservare che la matrice D calcolata come segue è diagonale:

$$D = MM^T$$

ovvero, moltiplicata per la sua trasposta dà luogo ad una matrice diagonale. Quindi la 5.2 può essere riscritta:

$$M^\dagger = M^T D^{-1}\,.$$

É possibile ora normalizzare la matrice M rispetto alla geometria del velivolo, introducendo:

$$\tilde{M} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ s_\alpha & -s_\alpha & -s_\alpha & s_\alpha & -s_\alpha & s_\alpha & s_\alpha & -s_\alpha \\ -c_\alpha & -c_\alpha & c_\alpha & c_\alpha & -c_\alpha & -c_\alpha & c_\alpha & c_\alpha \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{bmatrix}$$

$$M = K\tilde{M}$$

Dove K è una matrice diagonale che contiene i parametri dimensionali del sistema ed il coefficiente c definito precedentemente. La matrice pseudoinversa può dunque essere riscritta:

$$M^\dagger = M^T D^{-1} = \tilde{M}^T K^T D^{-1} \,. \tag{5.3}$$

Infine è possibile scrivere la relazione tra le velocità al quadrato dei motori e le forze generate in forma matriciale:

$$\mathbf{T}_m = k_t \boldsymbol{\Omega}^2 \tag{5.4}$$

dove k_t è il coefficiente di Thrust scalare. Dunque considerando le relazioni 5.4, 5.3, è possibile scrivere:

$$\boldsymbol{\Omega}^2 = k_t^{-1}\mathbf{T}_m = k_t^{-1} M^\dagger \mathbf{T}_\tau = k_t^{-1}\tilde{M}^T K D^{-1}\mathbf{T}_\tau \,.$$

In realtà il microcontrollore orginale del sistema utilizza la sola matrice $\tilde{M}$ per ricavare le forze da assegnare ai motori $\mathbf{T}_m$, poichè considera implicitamente nella taratura del controllo gli altri prodotti matriciali. Il software utilizza dunque degli indici di pseudo coppie e forze $\mathbf{T}_\tau^*$ in uscita dal sistema di controllo:

$$\boldsymbol{\Omega}^2 = \tilde{M}^T \mathbf{T}_\tau^* \,.$$

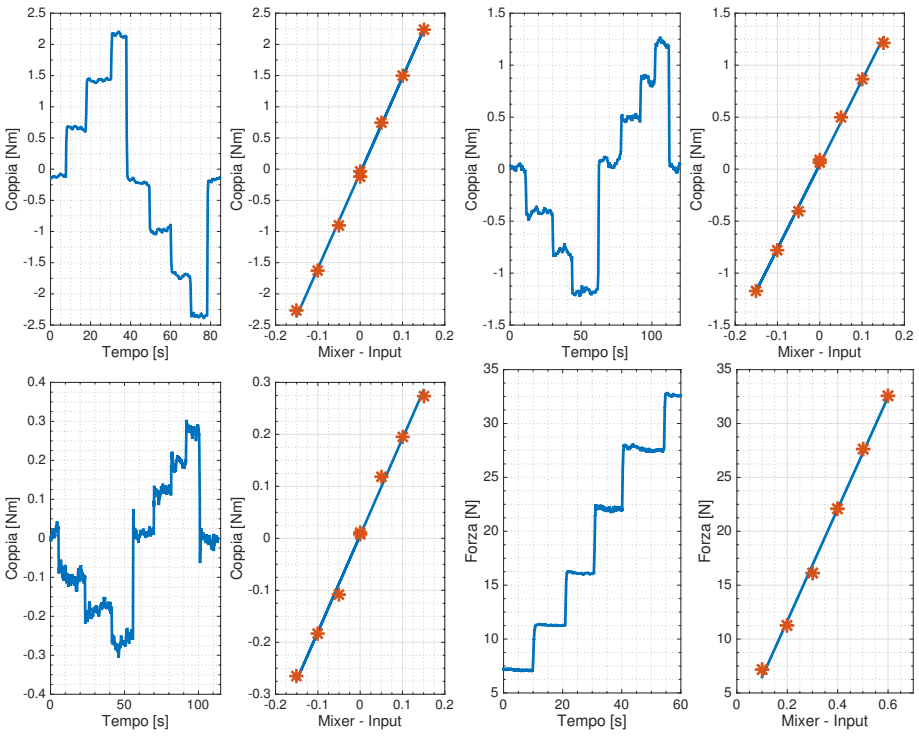
La relazione che intercorre tra $\mathbf{T}_\tau^*$ e $\mathbf{T}_\tau$ è dunque:

$$\mathbf{T}_\tau = M\,\mathbf{T}_m = M k_t\,\boldsymbol{\Omega}^{\mathbf{2}} = \; M k_t \tilde{M}^T \mathbf{T}_\tau^* = k_t\, M\tilde{M}^T \mathbf{T}_\tau^* = F\mathbf{T}_\tau^*$$

L'obiettivo della prova è stimare questa relazione, osservando che ciascun elemento $\mathbf{T}_\tau$ è funzione di un solo elemento di $\mathbf{T}_\tau^*$ poichè F è una matrice diagonale:

$$F = k_t\, M\tilde{M}^T = k_t\, M(K^{-1}M)^T = k_t\, MM^TK^{-1} = k_t\, DK^{-1}$$

Per effettuare la stima della relazione è necessario conoscere il valore delle variabili in ingresso e in uscita dalla stessa. Le variabili in ingresso sono note in quanto $\mathbf{T}_\tau^*$ possono essere imposte manualmente programmando il microcontrollore. Le variabili in uscita $\mathbf{T}_\tau$ possono essere misurate vincolando un sensore di forza a terra e alla base del drone. Si presentano i grafici rappresentanti le quattro prove effettuate per la stima delle quattro relazioni. Nei grafici è riportata la storia temporale di ciascuna prova, nelle figure di sinistra, e il legame stimato, nelle figure di destra.



Dai risultati ottenuti si nota che il legame tra $\mathbf{T}_\tau^*$ è, come ci si aspettava, approssimabile con una retta passante per l'origine. In particolare, analizzando i dati si ottengono coefficienti R^2 di determinazione di circa 0.99 approssimando la relazione con rette ai minimi quadrati passanti per

l'origine. Si riporta infine il valore della matrice stimata:

$$F = \begin{bmatrix} 52 & 0 & 0 & 0 \\ 0 & 15.31 & 0 & 0 \\ 0 & 0 & 6.16 & 0 \\ 0 & 0 & 0 & 1.86 \end{bmatrix}$$

5.3 Controllo di assetto del Drone

Si è testato il controllo di assetto del drone su un banco prova in grado di vincolare i GdL traslazionali, di imbardata e beccheggio del sistema. Si è quindi registrata la risposta dell'anello di controllo di assetto ad un gradino di riferimento di rollio.
Per fare ciò si è provveduto a vincolare opportunamente il drone ad una sbarra metallica orizzontale vincolata tramite cuscinetti. Il setup è visibile in fig. 5.1.

Figura 5.1: Banco di prova per testare il controllo di assetto.

Il controllore utilizzato è presentato in sezione 3.5 ed è composto da un regolatore detto P/PI che realizza un controllo in cascata sulla velocità e la posizione angolare del drone. Considerando la taratura effettuata sul sistema e caratterizzata da:

$$\omega_{cv} = K_{pv}/I_R = 25\,[\text{rad/s}]$$
$$\omega_z = K_{iv}/K_{pv} = \omega_{cv}/5 = 5\,[\text{rad/s}]$$
$$\omega_{cp} = \omega_{cv}/5 = 5\,[\text{rad/s}]$$

in cui il tempo di assestamento al 95% atteso dal sistema di controllo è di circa 0.6 s. La risposta al gradino ha restituito il grafico di fig. 5.2, dove si riporta la risposta reale, quella attesa, ed il riferimento imposto

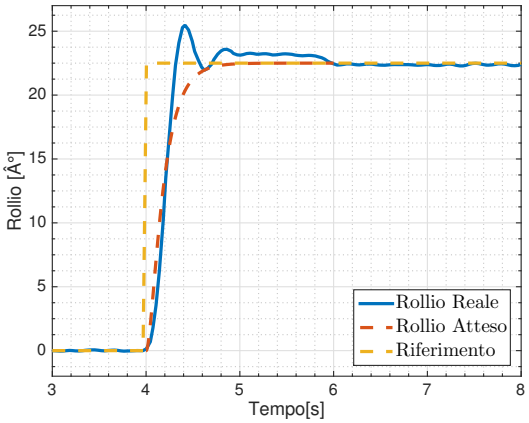


Figura 5.2: Risposta allo step dell'anello di controllo di rollio.

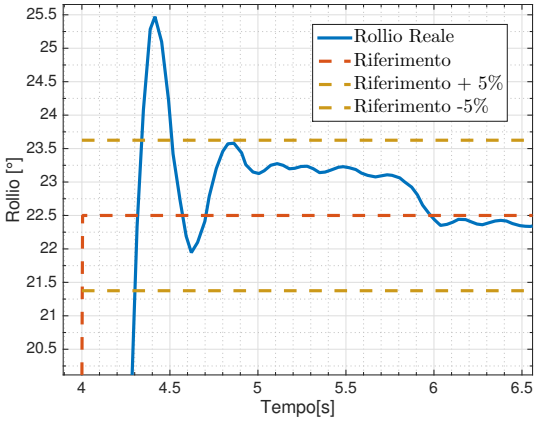


Figura 5.3: Ingrandimento della risposta al gradino dell'anelli di controllo di rollio.

all'angolo di rollio. Si notano delle differenze rispetto alla risposta attesa, in particolare si evidenzia, in fig. 5.3, una sovraelongazione e fenomeni non lineari che portano il sistema a stabilizzarsi inizialemente ad un valore superiore rispetto a quello imposto, che viene comunque raggiunto, sebbene con un certo ritardo. Queste non linearità sono probabilmente riconducibili alla presenza di attrito non trascurabile sul banco prova.

Osservando con più attenzione la risposta in prossimità dell'assestamento si nota in particolare che la sovraelongazione è pari a circa il 13% del valore del riferimento. Inoltre la risposta rimane entro $\pm 5\%$ del valore

finale dopo circa 0.5 s dall'invio del riferimento, riscontrando un risultato compatibile con il tempo di assestamento attteso. La risposta del sistema ha dato risultati comparabili con quelli attesi.

Il passo successivo è stato valutare la stabilità del controllo di assetto complessivo. Per effettuare questo test è stato necessario preparare un nuovo setup sperimentale, che permettesse di salvaguardare la sicurezza delle persone presenti alla prova nel caso di instabilizzazione del sistema. Nel dettaglio il drone è stato appeso tramite un cavo di acciaio che è poi stato condotto, attraverso due carrucole, a quattro metri di distanza ed è stato vincolato al terreno, in modo che in occasione di uno spegnimento accidentale dei motori il drone non potesse toccare terra. Il setup è raffigurato in fig. ??, dove vengono schematizzate le posizioni delle carrucole. Per verificare la stabilità dell'anello di controllo d'assetto, il drone è stato

Figura 5.4: Setup sperimentale per test di volo.

guidato manualmente (tramite telecomando) e portato in condizione di *hovering* utilizzando per il controllo di assetto la taratura sopra descritta. Infine per testare la risposta dell'anello di imbardata si è utilizzata la stima

di una telecamera posta sul drone ed in grado, tramite analisi di immagine, di riconoscere un marker posto in prossimità del dorne, ricavando in questo modo la posizione di imbardata relativa. In fig. 5.5 è visibile la telecamera posta sulla parte anteriore del drone utilizzata per la stima. Durante la

Figura 5.5: Condizione del drone durante la prova.

prova il drone è stato sollevato a circa $1,3$ m in modalità manuale per evitare i disturbi relativi al cavo d'acciaio ed in condizione di *hovering* è stato inviato un riferimento a gradino di imbardata. La risposta ottenuta dal sistema è riportata in fig. 5.6. Si nota nuovamente che il sistema raggiunge il gradino di riferimento imposto, anche se si notano differenze rispetto alla risposta attesa. Queste differenze sono probabilmente riconducibili alla presenza di disturbi aerodinamici e a errori di stima del modello.
In particolare si può osservare che il tempo di ritardo reale, ovvero il tempo per raggiungere il 50% del valore desiderato, ed il tempo di ritado reale coincidono, mentre il tempo di assestamento al 95% risulta essere in ritardo di circa 0.55 s (fig. 5.7).

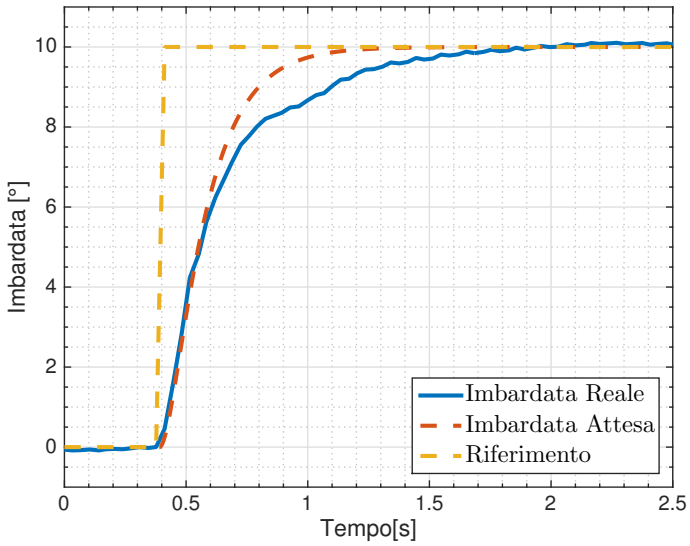


Figura 5.6: Risposta al gradino dell'anello di controllo d'imbardata.

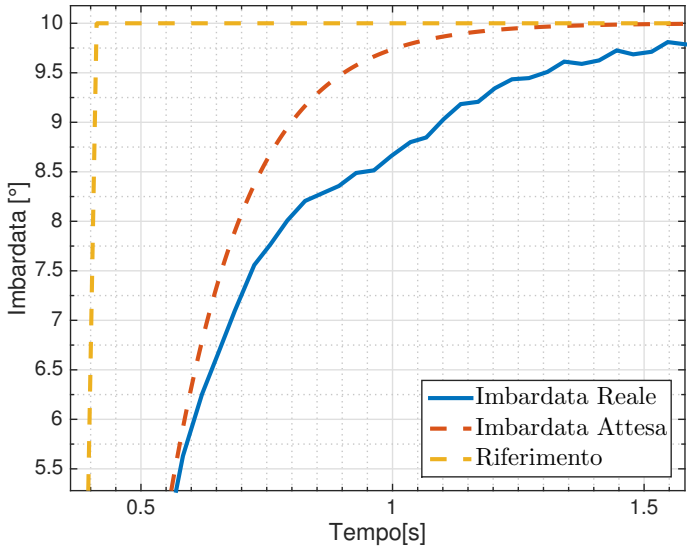


Figura 5.7: Risposta al gradino dell'anello di controllo d'imbardata.

5.4 Controllo di posizione del Drone

Dopo aver stabilizzato il sistema si è testato il controllo di posizione nello spazio cartesiano. La tipologia di controllo utilizzata per il controllo di posizione è del tipo P/PI (sez. 3.5) su tutti e tre i gradi di libertà traslazionali.
Il setup sperimentale utilizzato è stato lo stesso presentato per la prova di stabilità del controllo d'assetto complessivo. La stima della posizione è stata invece effettuata sempre utilizzando la telecamera posta sul drone in grado, tramite analisi di immagine, di riconoscere un marker.
In primo luogo si è testato il controllo di posizione in direzione z, o controllo di altitudine, che permette il raggiungimento della condizione di *hovering* del drone. Tale condizione è fondamentale per permettere il controllo di posizione sul piano xy.
Il controllo di altitudine è stato tarato con i seguenti valori:

$$
\begin{aligned}
f_N &= \pi f_c = 62,8\,[\text{rad/s}] \\
\omega_{cv} &= K_{pv}/m_q = 2.5\,[\text{rad/s}] \\
\omega_z &= K_{iv}/K_{pv} = \omega_{cv}/5 = 0.5\,[\text{rad/s}] \\
\omega_{cp} &= \omega_{cv}/5 = 0.5\,[\text{rad/s}]
\end{aligned}
$$

dove m_q è il valore della massa del drone, f_N è la frequenza di Nyquist del sistema, ω_{cv} è la banda passante dell'anello di velocità, ω_z è la frequenza a cui poniamo lo zero del controllore e ω_{cp} è la banda passante dell'anello di posizione. Ci si aspetta dunque un valore del tempo di assestamento al 99% del riferimento di 10 s. La prova è stata effettuata portando il drone in posizione di *hovering* tramite controllo manuale, in modo che il cavo di sicurezza non influisse sulla prova. In questa condizione è stato attivato il controllo di posizione ed è stato inviato un riferimento a un gradino di 30 cm. La risposta è riportata in fig. 5.8.

Si nota che il sistema si avvicina al riferimento di posizione assegnatogli, anche se soggetto a numerosi disturbi che ne perturbano l'andamendo. In questo caso è più complicato stimare un tempo di assestamento del sistema

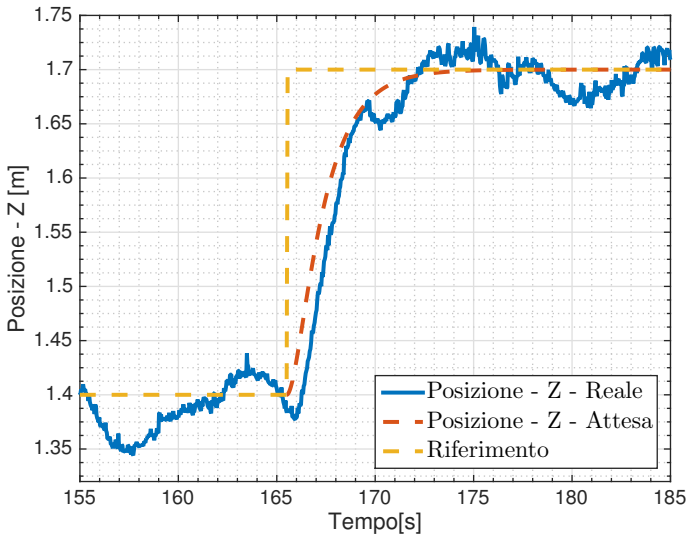


Figura 5.8: Risposta al gradino dell'anello di controllo di altitudine

a causa delle numerose oscillazioni intorno alla posizione raggiunta. Queste sono probabilmente causate dai disturbi aerodinamici a cui il sistema è sottoposto. Si osserva comunque che le oscillazioni sono dell'ordine dei 10 cm intorno alla posizione di riferimento e che il sistema segue, sebbene soggetto a disturbi, il riferimento inviato.

In secondo luogo si testa il controllo di posizione sul piano xy. Il setup sperimentale utilizzato è lo stesso già esposto per la prova del controllo di altitudine. La taratura del controllo di posizione in direzione xy deve però tenere conto anche dei limiti imposti dall'anello di controllo di assetto (3.5). Si riporta la taratura utilizzata:

$$
\begin{aligned}
f_N &= \pi f_c = 62,8\,[\text{rad/s}] \\
\omega_{cp,\vartheta} &= 5\,[\text{rad/s}] \\
\omega_{cv} &= K_{pv}/m_q = 1.5\,[\text{rad/s}] \\
\omega_z &= K_{iv}/K_{pv} = \omega_{cv}/5 = 0.3\,[\text{rad/s}] \\
\omega_{cp} &= \omega_{cv}/5 = 0.3\,[\text{rad/s}]
\end{aligned}
$$

dove f_N è la frequenza di Nyquist del sistema, $\omega_{cp,\vartheta}$ è la banda passante dell'anello di assetto, ω_{cv} è la banda passante dell'anello di velocità, m_q

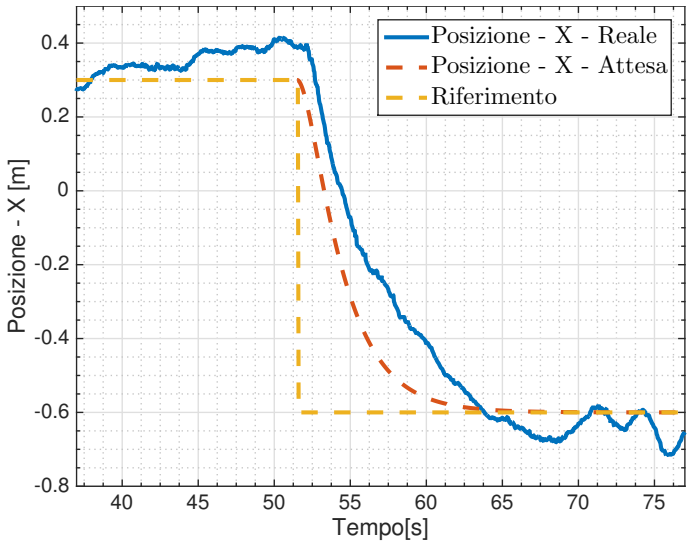


Figura 5.9: Risposta al gradino dell'anello di controllo di posizione.

è il valore della massa del drone, ω_z è la frequenza a cui poniamo lo zero del nostro controllore e ω_{cp} è la banda passante dell'anello di posizione. Il tempo di assestamento atteso è dunque nell'ordine dei 20 s.
La risposta ad uno step di posizione in direzione x è presentato in fig. 5.9.

Si nota anche in questo caso che il sistema raggiunge il riferimento di posizione assegnatogli sebbene soggetto a numerosi disturbi, in maniera analoga a quanto già osservato nella prova del controllo di altitudine.

5.5 Controllo MPC applicato al solo braccio robotico

Si presentano in questa sezione i risultati sperimentali ottenuti implementanto il controllo di alto livello sul solo braccio robotico. La prova è stata effettuata vincolando il braccio al di sotto del drone ed implementando un *nodo* ROS in grado di leggere le posizioni dei giunti dagli encoder, ottmizzare una funzione di costo contenente la cinematica del solo braccio, ed inviare i riferimenti di posizione e velocità dei giunti al controllore dei Dynamixel, ovvero i motori che controllano i giunti. Il modello per la predizione utilizzato in ROS è quello che descrive la cinematica del solo

braccio. Nel dettaglio il modello utilizzato per la predizione è lo stesso presentato in 3.4.3, ma senza considerare la cinematica del drone:

$$\dot{\mathbf{q}} \;=\; \mathbf{u_q}\,,$$

dove ${}^{i}\mathbf{q}_b$ è la posizione del drone, mentre ${}^{i}\mathbf{v}_b$ è la velocità dello stesso. Il vettore di stato e degli ingressi sono dunque:

$$\begin{aligned}\mathbf{x} &= \mathbf{q}\\ \mathbf{u} &= \mathbf{q_u}\,,\end{aligned}$$

mentre come funzione di costo da minimizzare si sceglie:

$$H(\mathbf{x}(k),\cdot,k) \;=\; \sum_{i=0}^{N-1}(||\mathbf{y}^o(k+i) - {}^{b}\,\mathbf{p}_e(\mathbf{x}(k+i))||_Q^2 + ||\mathbf{u}(k+i)||_R^2)\,,$$

dove ${}^{b}\mathbf{p}_e(\mathbf{x}(k+i))$ rappresenta la posizione dell'*end-effector* in funzione delle variabili di giunto del manipolatore, nel sistema di riferimento E_b. $\mathbf{y}^o(k+i)$ rappresenta la posizione desiderata per l'*end-effector*, mentre $\mathbf{u}(k+i)$ rappresenta le variabili di controllo. In particolare ${}^{b}\mathbf{p}_e(\mathbf{x}(k+i))$ è così strutturata:

$${}^{b}\mathbf{p}_e = \begin{bmatrix} a_2\cos(q_1)\cos(q_2) - d_4(\cos(q_1)\cos(q_2)\sin(q_3) + \cos(q_1)\sin(q_2)\cos(q_3)) \\ a_2\sin(q_1)\cos(q_2) - d_4(\sin(q_1)\cos(q_2)\sin(q_3) + \sin(q_1)\sin(q_2)\cos(q_3)) \\ d_1 - d_4(\cos(q_2)\cos(q_3) - \sin(q_2)\sin(q_3)) - a_2\sin(q_2) \end{bmatrix} \tag{5.5}$$

dove q_i è la i-esima variabile di giunto del braccio robotico, mentre a_2, d_1 e d_4 sono i parametri geometrici dimensionali dei diversi *link*.

Per quanto riguarda le matrici utilizzate nel funzionale di costo, sono state scelte le seguenti dopo alcune prove:

$$\begin{aligned} Q &= \mathrm{diag}\begin{bmatrix}1 & 1 & 1\end{bmatrix}\\ R &= \mathrm{diag}\begin{bmatrix}0.1 & 0.1 & 0.1\end{bmatrix}\,. \end{aligned}$$

Sono inoltre stati considerati dei vincoli sulle velocità massime assegnabili come riferimenti, ovvero 1 rad/s per ciascun giunto. La pubblicazione dei riferimenti avviene ad una frequenza di 2 Hz, una frequenza che permette al sistema di controllo di basso livello del braccio di seguire i riferimenti di velocità inviati. L'anello di basso livello è costituito da un regolatore P/PI presentato in 3.5, e tarato con i seguenti parametri:

$$\begin{aligned}\omega_{cv} &= K_{pv}/I_m = 20\,[\text{rad/s}]\\ \omega_z &= K_{iv}/K_{pv} = \omega_{cv}/10 = 2\,[\text{rad/s}]\\ \omega_{cp} &= \omega_{cv}/10 = 2\,[\text{rad/s}]\end{aligned}$$

dove ω_{cv} è la banda passante dell'anello di velocità, ω_z è la frequeza a cui poniamo lo zero del regolatore e ω_{cp} è la banda passante dell'anello di posizione.

La prova effettuata considera come target il raggiungimento di due punti per l'*end-effector* di coordinate:

$$\begin{aligned}\mathbf{y}_1^o &= \begin{bmatrix}0.1 & 0.1 & 0.2\end{bmatrix}\\ \mathbf{y}_2^0 &= \begin{bmatrix}0.2 & -0.2 & 0.3\end{bmatrix}.\end{aligned}$$

La prova effettutata è raffigurata in fig. 5.10 e fig. 5.11. In fig. 5.10 sono rappresentati gli andamenti delle posizioni dell'*end-effector* nel tempo, mentre in fig. 5.11 sono raffigurati gli andamenti delle variabili di giunto e i riferimenti generati dal sistema di alto livello. In primo luogo si osserva che il sistema raggiunge i riferimenti generati dal controllo di alto livello, e che le posizioni desiderate nella terna E_b vengono raggiunti. L'errore che si ottiene sulla coordinata y prima delll'invio del secondo *set-point* deriva probabilmente da una compensazione non ottimale di disturbi (attriti, gravità). In secondo luogo si osserva la presenza di alcune sovraelongazioni prima del raggiungimento dei *set-point* nella terna E_b. Queste sovraelonga-

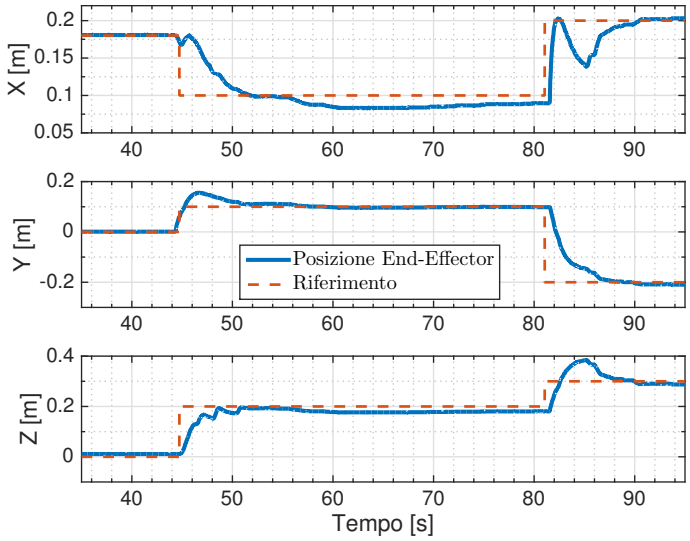


Figura 5.10: Andamento della risposta dell'*end-effector* nelle coordinate xyz.

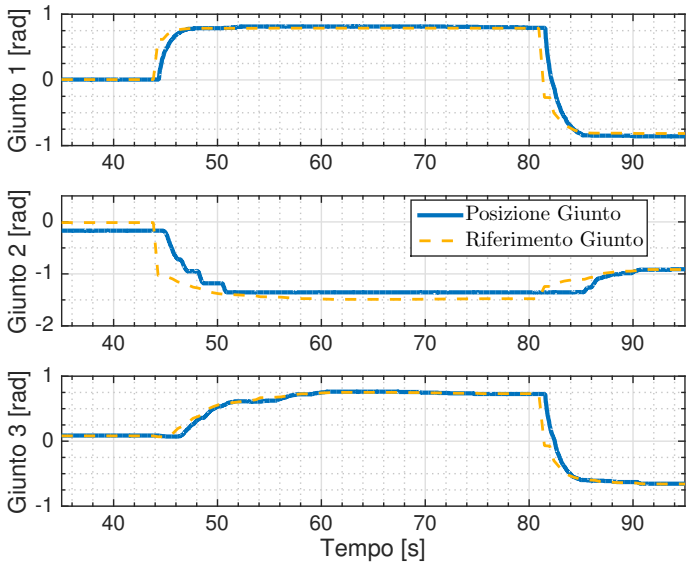


Figura 5.11: Riferimenti generati e risposta delle variabili di giunto.

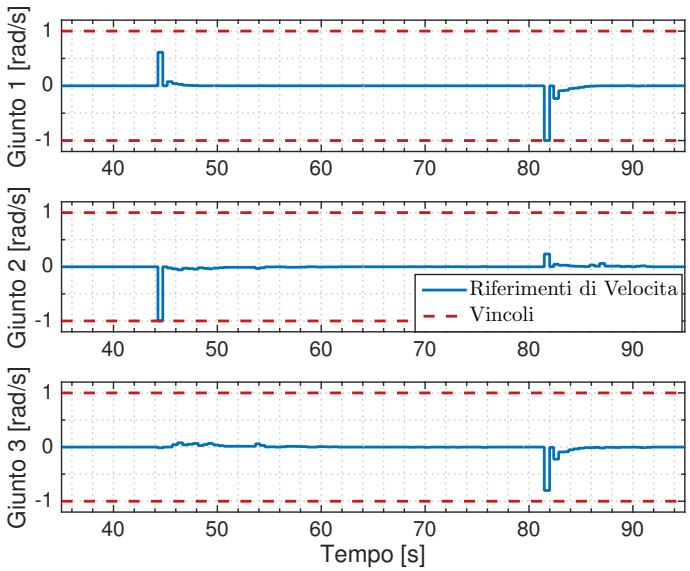


Figura 5.12: Vincoli sull'attuazioni.

zioni derivano dal contemporaneo movimento dei diversi giunti del braccio, accoppiati all'interno della funzione cinematica non lineare.

Per quanto riguarda il sistema di controllo di alto livello, si nota come, a fronte di un riferimento a gradino, esso generi dei riferimenti più dolci per il sistema di controllo di basso livello. Si osserva infine che i vincoli imposti per il sistema vengono rispettati, infatti i riferimenti di velocità generati sono sempre inferiori all'unità (fig, 5.12). Si riportano infine due fotografie scattate al termine della prova (5.13, fig. 5.14), dove il braccio pone l'*end-effector* in posizione [0.2 -0.2 0.3].

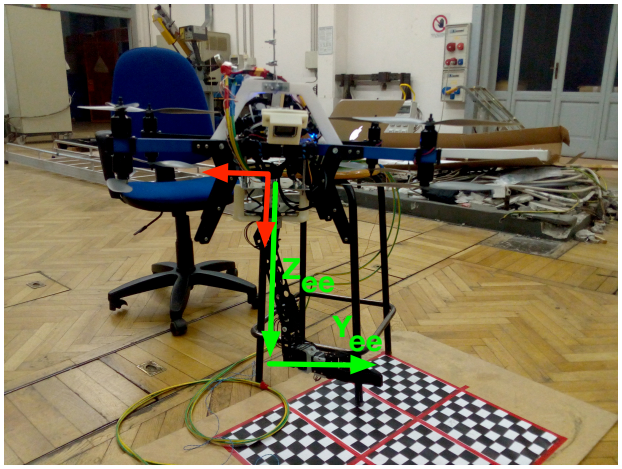


Figura 5.13: Posizione dell'*end-effector* al termine della prova.

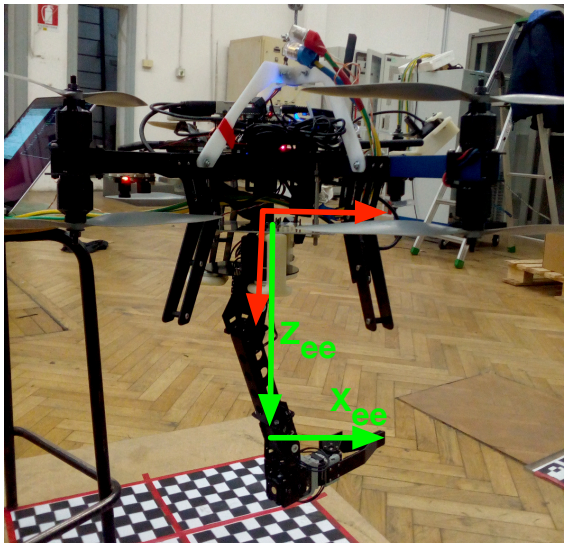


Figura 5.14: Posizione dell'*end-effector* al termine della prova.

Conclusioni

La presente tesi si proponeva come obiettivo principale lo sviluppo e l'applicazione si un algoritmo di controllo che permettesse l'inseguimento di una traiettoria da parte dell'*end-effector* del manipolatore aereo. In particolare il problema è stato risolto tramite l'applicazione di tecniche di controllo predittivo. Per raggiungere questo obiettivo sono stati sviluppati tre diversi modelli di predizione: un modello interamente dinamico, un modello ibrido ed un modello cinematico. I tre modelli sono stati confrontati dal punto di vista della complessità computazionale.

Il controllo predittivo che utilizza il modello completamente cinematico, richiedendo tempi computazionali inferiori, è stato utilizzato per introdurre un obiettivo secondario: ridurre il disallineamento tra baricentro del braccio robotico e del drone. L'obiettivo secondario è stato introdotto con una priorità inferiore, ovvero viene otimizzato solo se non influenza l'andamento del principale.

Parte del sistema di controllo è stato inotre testato tramite simulazioni *Software In The Loop*, implementando l'algoritmo di controllo sulla piattaforma ROS e simulando il sistema tramite il simulatore Gazebo.

Infine sono stati studiati i sistemi di controllo di basso livello. Sono stati testati in simulazione Matlab, realizzati in ROS, ed infine validati tramite prove sperimentali. I test hanno permesso, in particolare, di validare i sistemi di controllo di basso livello e dell'algoritmo di alto livello applicato al solo braccio robotico.

5.6 Sviluppi futuri

Il presente lavoro lascia alcune porte aperte che possono essere intraprese per migliorare i risultati ottenuti.

In primo luogo, dal punto di vista simulativo, si può pensare di ottimizzare le tempistiche utilizzando solutori ottimizzati per problemi non lineari, in modo da utilizzare i modelli contenenti la dinamica per la predizione con più efficacia. Anche le simulazioni che utilizzano come modello predittivo la sola cinematica del robot possono essere soggette a sviluppi implementando migliori tecniche per l'inseguimento delle traiettorie. Per quanto riguarda poi l'approccio gerarchico, è possibile introdurre altri obiettivi secondari, ad esempio un indice di manipolabilità.

In secondo luogo si può pensare di migliorare le simulazioni condotte in Gazebo, realizzando un modello del sistema complessivo ed eventualmente aggiungendo al modello virtuale i sensori realmente utilizzati.

Infine, per quanto riguarda le prove sperimentali, si può pensare di implementare l'algoritmo sul sistema complessivo.

Appendice A

Taratura anelli di controllo di basso livello

A.1 Anello di controllo di assetto e posizione del drone

Si riporta la procedura di taratura del sistema di controllo di basso livello del drone descritta in sez. 3.5.

In primo luogo presentiamo la procedura di taratura per il sistema di controllo di assetto. Questo viene sintetizzato realizzando un controllo in cascata tra posizione e velocità angolare. Supponendo di avere una frequenza di campionamento di 200 Hz (coincidente con quella reale), si può tarare l'anello interno di velocità:

$$
\begin{aligned}
f_N &= \pi f_c = 628\,[\text{rad/s}]\,, \\
\omega_{cv,\vartheta} &= K_{pv}/I_R = f_N/10 \simeq 60\,[\text{rad/s}]\,, \\
\omega_z &= K_{iv}/K_{pv} = \omega_{cv}/3 \simeq 20\,[\text{rad/s}]\,,
\end{aligned}
$$

dove f_N è la frequenza di Nyquist del sistema, $\omega_{cv,\vartheta}$ è la banda passante dell'anello di velocità angolare, ω_z è la frequenza a cui poniamo lo zero del

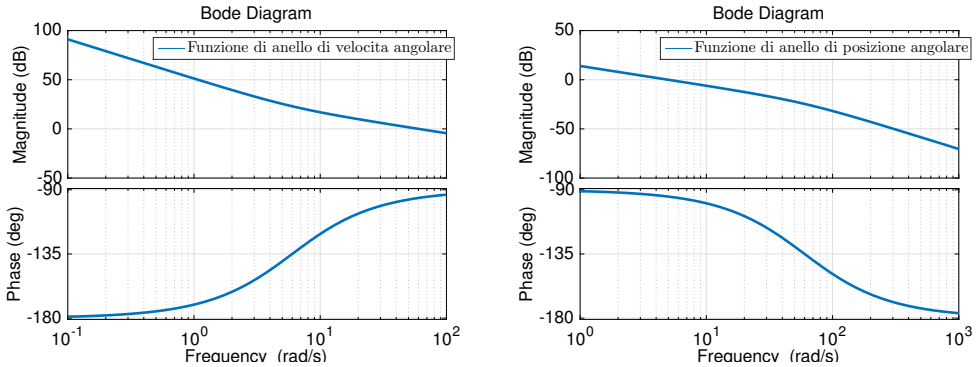


Figura A.1: Diagramma di Bode: anelli per il controllo della posizione angolare.

nostro regolatore e I_R è il momento d'inerzia del sistema in esame.

A questo punto ci occupiamo della taratura dell'anello di posizione angolare. Essa deve tenere conto della presenza di un anello di velocità interno che deve essere sufficientemente più veloce. Si sceglie dunque:

$$\omega_{cp,\vartheta} = \omega_{cv,\vartheta}/10 = 6\,[\text{rad/s}]\,,$$

dove $\omega_{cp,\vartheta}$ è la banda passante dell'anello di posizione angolare.
In secondo luogo tariamo l'anello di controllo di posizione cartesiana del drone. Anche questo controllore è composto dai due anelli di velocità e posizione in cascata. I valori scelti per l'anello di velocità sono i seguenti:

$$\begin{aligned}
&\omega_{cp,\vartheta} = 6\,[\text{rad/s}]\,,\\
&\omega_{cv,xy} = K_{pv}/m_q = \omega_{cp,\vartheta}/3 = 2\,[\text{rad/s}]\,,\\
&\omega_z = K_{iv}/K_{pv} = \omega_{cv,xy}/10 = 0.2\,[\text{rad/s}]\,,
\end{aligned}$$

dove $\omega_{cp,\vartheta}$ è la banda passante dell'anello di posizione angolare, $\omega_{cv,xy}$ è la banda passante dell'anello di posizione xy, ω_z è la frequenza a cui poniamo lo zero del nostro regolatore e infine m_q è la massa del drone.

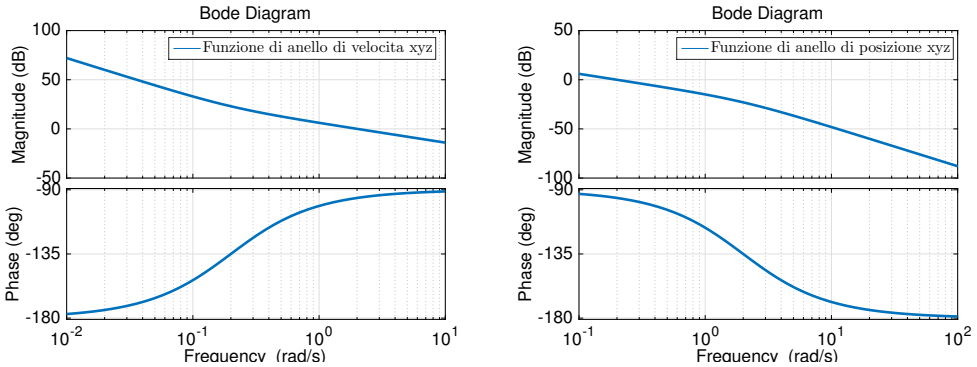


Figura A.2: Diagramma di Bode: anelli per il controllo della posizione xy.

Si passa infine a tarare l'anello di controllo in posizione xy:

$$\omega_{cp,xy} = K_{p,xy} = \omega_{cv,xy}/10 = 0.2\,[\text{rad/s}]\,,$$

dove $\omega_{cp,xy}$ è la banda passante dell'anello di posizione xy, $\omega_{cv,xy}$ è la banda passante dell'anello di velocità. Per la coordinata z (controllo di altitudine) si è proceduto allo stesso modo sintetizzando un regolatore con le stesse caratteristiche di prontezza.

A.2 Anello di controllo del braccio robotico

Si presenta infine la procedura di taratura del controllore di basso livello del braccio robotico. Si tratta dello stesso tipo di regolatore realizzato per il controllo di posizione del drone, si segue dunque la stessa procedura di taratura, tenendo conto dell'inerzia motore e imponendo una banda passante dipendente dai limiti imposti dalla frequenza di campionamento del sistema. Si suppone di avere una frequenza di campionamento di 50

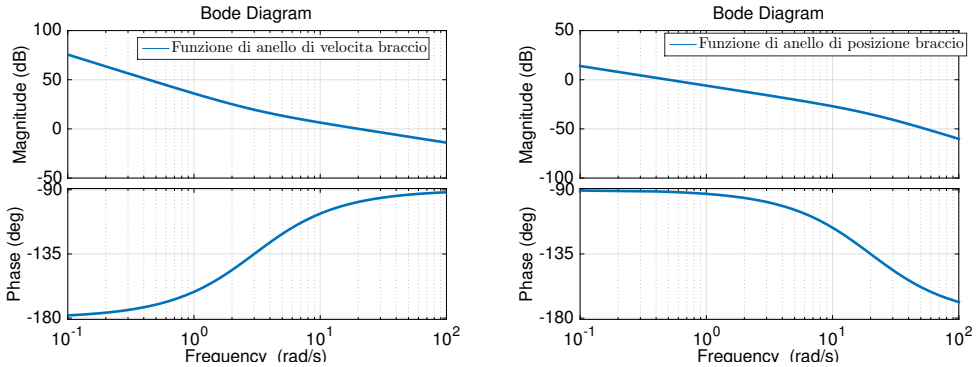


Figura A.3: Diagrammi di Bode: anelli per il controllo della posizione dei giunti.

Hz (come da sistema reale), e si impone dunque:

$$
\begin{aligned}
f_N &= \pi f_c = 157\,[\text{rad/s}] \\
\omega_{cv} &= K_{pv}/I_m = 30\,[\text{rad/s}] \\
\omega_z &= K_{iv}/K_{pv} = \omega_{cv}/10 = 3\,[\text{rad/s}] \\
\omega_{cp} &= K_p = \omega_{cv}/5 = 6\,[\text{rad/s}]
\end{aligned}
$$

dove f_N è la frquenza di Nyquist, ω_{cv} è la banda passante dell'anello di velocità, I_m è l'inerzia del motore, ω_z è la frequenza a cui poniamo lo zero del nostro regolatore e infine ω_{cp} è la banda passante dell'anello di posizione.

Bibliografia

[1] S. Bouabdallah, A. Noth e R. Siegwart. «PID vs LQ control techniques applied to an indoor micro quadrotor». In: *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. 2004, 2451–2456 vol.3. DOI: 10.1109/IROS.2004.1389776 (cit. a p. 4).

[2] Samir Bouabdallah, Pierpaolo Murrieri e Roland Siegwart. «Towards autonomous indoor micro VTOL». In: *Autonomous Robots* 18.2 (2005), pp. 171–183 (cit. a p. 4).

[3] Gabriel M Hoffmann et al. «Quadrotor helicopter flight dynamics and control: Theory and experiment». In: *Proc. of the AIAA Guidance, Navigation, and Control Conference*. Vol. 2. 2007 (cit. a p. 4).

[4] Jinok Shin et al. «Model-based optimal attitude and positioning control of small-scale unmanned helicopter». In: *Robotica* 23.01 (2005), pp. 51–63 (cit. a p. 4).

[5] Jonathan P How et al. «Real-time indoor autonomous vehicle test environment». In: *Control Systems, IEEE* 28.2 (2008), pp. 51–64 (cit. a p. 4).

[6] M La Civita et al. «Design and Flight Testing of an H00 Controller for a Robotic Helicopter». In: *Journal of Guidance, Control, and Dynamics* 29.2 (2006), pp. 485–494 (cit. a p. 4).

[7] M Bouchoucha et al. «Integral backstepping for attitude tracking of a quadrotor system». In: *Elektronika ir Elektrotechnika* 116.10 (2011), pp. 75–80 (cit. a p. 4).

[8] Z.T. Dydek, A.M. Annaswamy e E. Lavretsky. «Adaptive Control of Quadrotor UAVs: A Design Trade Study With Flight Evaluations». In: *Control Systems Technology, IEEE Transactions on* 21.4 (2013), pp. 1400–1406. ISSN: 1063-6536. DOI: 10.1109/TCST.2012.2200104 (cit. a p. 4).

[9] Rong Xu e U. Ozguner. «Sliding Mode Control of a Quadrotor Helicopter». In: *Decision and Control, 2006 45th IEEE Conference on*. 2006, pp. 4957–4962. DOI: 10.1109/CDC.2006.377588 (cit. a p. 4).

[10] Mahyar Abdolhosseini, YM Zhang e Camille Alain Rabbath. «An efficient model predictive control scheme for an unmanned quadrotor helicopter». In: *Journal of intelligent & robotic systems* 70.1-4 (2013), pp. 27–38 (cit. a p. 4).

[11] Konstantin Kondak et al. «Closed-loop behavior of an autonomous helicopter equipped with a robotic arm for aerial manipulation tasks». In: *International Journal of Advanced Robotic Systems* 10 (2013) (cit. a p. 4).

[12] Matko Orsag et al. «Stability control in aerial manipulation». In: *American Control Conference (ACC), 2013*. IEEE. 2013, pp. 5581–5586 (cit. a p. 4).

[13] Dario Fusato, Giorgio Guglieri e Roberto Celi. «Flight dynamics of an articulated rotor helicopter with an external slung load». In: *Journal of the American Helicopter Society* 46.1 (2001), pp. 3–13 (cit. a p. 4).

[14] ARCAS. «http://www.arcas-project.eu.» In: () (cit. a p. 5).

[15] *AEROARMS* (cit. a p. 5).

[16] Justin Thomas et al. «Avian-inspired grasping for quadrotor micro uavs». In: *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers. 2013, V06AT07A014–V06AT07A014 (cit. a p. 5).

[17] G Arleo et al. «Control of quadrotor aerial vehicles equipped with a robotic arm». In: *Control & Automation (MED), 2013 21st Mediterranean Conference on*. IEEE. 2013, pp. 1174–1180 (cit. a p. 5).

[18] Vincenzo Lippiello e Fabio Ruggiero. «Exploiting redundancy in Cartesian impedance control of UAVs equipped with a robotic arm». In: *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE. 2012, pp. 3768–3773 (cit. a p. 5).

[19] A. Khalifa et al. «Modeling and control of a new quadrotor manipulation system». In: *Innovative Engineering Systems (ICIES), 2012 First International Conference on*. 2012, pp. 109–114. DOI: 10.1109/ICIES.2012.6530854 (cit. a p. 5).

[20] W. Decré, H. Bruyninckx e J. De Schutter. «Extending the iTaSC Constraint-based Robot Task Specification Framework to Time-Independent Trajectories and User-Configurable Task Horizons». In: *Robotics and Automation (ICRA), 2013 IEEE International Conference on*. 2013, pp. 1941–1948. DOI: 10.1109/ICRA.2013.6630835 (cit. a p. 5).

[21] Martin de Lasa e Aaron Hertzmann. «Prioritized Optimization for Task-space Control». In: *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IROS'09. St. Louis, MO, USA: IEEE Press, 2009, pp. 5755–5762. ISBN: 978-1-4244-3803-7. URL: http://dl.acm.org/citation.cfm?id=1732643.1732981 (cit. a p. 5).

[22] Claude Samson, Bernard Espiau e Michel Le Borgne. *Robot Control: The Task Function Approach*. Oxford University Press, 1991. ISBN: 0198538057 (cit. a p. 6).

[23] MW Chen e AMS Zalzala. «Dynamic modelling and genetic-based trajectory generation for non-holonomic mobile manipulators». In: *Control Engineering Practice* 5.1 (1997), pp. 39–48 (cit. a p. 6).

[24] F. Abdessemed. «Trajectory generation for mobile manipulators using a learning method». In: *Control Automation, 2007. MED '07.*

Mediterranean Conference on. 2007, pp. 1–6. DOI: 10.1109/MED.2007.4433659 (cit. a p. 6).

[25] Y Bouktir, M Haddad e T Chettibi. «Trajectory planning for a quadrotor helicopter». In: *Control and Automation, 2008 16th Mediterranean Conference on*. IEEE. 2008, pp. 1258–1263 (cit. a p. 6).

[26] Daniel Mellinger e Vijay Kumar. «Minimum snap trajectory generation and control for quadrotors». In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE. 2011, pp. 2520–2525 (cit. a p. 6).

[27] Markus Hehn e Raffaello D'Andrea. «Quadrocopter trajectory generation and control». In: *IFAC world congress*. Vol. 18. 1. 2011, pp. 1485–1491 (cit. a p. 6).

[28] Ivana Palunko, Rafael Fierro e Pedro Cruz. «Trajectory generation for swing-free maneuvers of a quadrotor with suspended payload: A dynamic programming approach». In: *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE. 2012, pp. 2691–2697 (cit. a p. 6).

[29] Gowtham Garimella e Marin Kobilarov. «Towards model-predictive control for aerial pick-and-place». In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 4692–4697 (cit. a p. 6).

[30] A. Del Prete et al. «Prioritized Optimal Control». In: *ArXiv e-prints* (ott. 2014). arXiv: 1410.4414 [cs.RO] (cit. a p. 7).

[31] Yoshiihiko Nakamura, Hideo Hanafusa e Tsuneo Yoshikawa. «Task-priority Based Redundancy Control of Robot Manipulators». In: *Int. J. Rob. Res.* 6.2 (lug. 1987), pp. 3–15. ISSN: 0278-3649. DOI: 10.1177/027836498700600201. URL: http://dx.doi.org/10.1177/027836498700600201 (cit. a p. 7).

[32] B. Siciliano e J. J. E. Slotine. «A general framework for managing multiple tasks in highly redundant robotic systems». In: *Advanced Robotics, 1991. 'Robots in Unstructured Environments', 91 ICAR.,*

Fifth International Conference on. 1991, 1211–1216 vol.2. DOI: 10.1109/ICAR.1991.240390 (cit. a p. 7).

[33] O. Kanoun, F. Lamiraux e P. B. Wieber. «Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task». In: *IEEE Transactions on Robotics* 27.4 (2011), pp. 785–792. ISSN: 1552-3098. DOI: 10.1109/TRO.2011.2142450 (cit. a p. 7).

[34] A. Santamaria-Navarro, V. Lipiello e J. Andrade-Cetto. «Task priority control for aerial manipulation». In: 2014, pp. 1–6. DOI: 10.1109/SSRR.2014.7017672 (cit. a p. 7).

[35] Vincenzo Lippiello et al. «Hybrid Visual Servoing with Hierarchical Task Composition for Aerial Manipulation». In: (2015) (cit. a p. 7).

[36] *3DR* (cit. a p. 13).

[37] *CrustCrawler Robotics* (cit. a p. 14).

[38] *ETH* (cit. a p. 15).

[39] Lorenz Meier, Dominik Honegger e Marc Pollefeys. «PX4: A node-based multithreaded open source robotics framework for deeply embedded platforms». In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 6235–6240 (cit. a p. 15).

[40] *Odroid XU3* (cit. a p. 16).

[41] Morgan Quigley et al. «ROS: an open-source Robot Operating System». In: *ICRA workshop on open source software* 3.3.2 (2009), p. 5 (cit. a p. 17).

[42] B. Siciliano et al. *Robotics: Modelling, Planning and Control*. Advanced Textbooks in Control and Signal Processing. Springer London, 2010. ISBN: 9781846286414. URL: https://books.google.it/books?id=jPCAFmE-logC (cit. alle pp. 21, 29).

[43] R. Mahony, V. Kumar e P. Corke. «Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor». In: *IEEE Robotics Automation Magazine* 19.3 (2012), pp. 20–32. ISSN: 1070-9932. DOI: 10.1109/MRA.2012.2206474 (cit. a p. 26).

[44] R.W. Prouty. *Helicopter performance, stability, and control*. PWS Engineering, 1986. ISBN: 9780534063603. URL: https://books.google.it/books?id=iLtTAAAAMAAJ (cit. a p. 27).

[45] L. Magni e R. Scattolini. *Complementi di controlli automatici*. Pitagora, 2006. ISBN: 9788837116477. URL: https://books.google.it/books?id=AXL_AQAACAAJ (cit. alle pp. 33, 42, 44).

[46] *Peter Corke* (cit. a p. 57).

[47] *NMPC, theory and algorithms* (cit. a p. 57).

[48] Nathan Koenig e Andrew Howard. «Design and use paradigms for gazebo, an open-source multi-robot simulator». In: *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*. Vol. 3. IEEE, pp. 2149–2154 (cit. a p. 67).

[49] Lorenz Meier et al. «PIXHAWK: A micro aerial vehicle design for autonomous flight using onboard computer vision». In: *Autonomous Robots* 33.1-2 (2012), pp. 21–39 (cit. a p. 68).

[50] B. Houska, H.J. Ferreau e M. Diehl. «ACADO Toolkit – An Open Source Framework for Automatic Control and Dynamic Optimization». In: *Optimal Control Applications and Methods* 32.3 (2011), pp. 298–312 (cit. a p. 69).

[51] *Optical Flow* (cit. a p. 13).

[52] Alberto Bemporad, Carlo A Pascucci e Claudio Rocchi. «Hierarchical and hybrid model predictive control of quadcopter air vehicles». In: *Analysis and Design of Hybrid Systems*. Vol. 3. 1. 2009, pp. 14–19.

[53] R. Quirynen, S. Gros e M. Diehl. «Efficient NMPC for nonlinear models with linear subsystems». In: *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. 2013, pp. 5101–5106. DOI: 10.1109/CDC.2013.6760690.

[54] Giovanni Massimo Buizza Avanzini. «Control strategies for redundant and mobile robotic manipulators subject to multiple constraints». In: Politecnico di Milano. 2015.

[55] A. M. Zanchettin e P. Rocco. «Near time-optimal and sensor-based motion planning for robotic manipulators». In: *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. 2013, pp. 965–970. DOI: 10.1109/CDC.2013.6760007.

[56] S Joe Qin e Thomas A Badgwell. «An overview of industrial model predictive control technology». In: *AIChE Symposium Series* 93.316 (1997), pp. 232–256.

[57] S Joe Qin e Thomas A Badgwell. «A survey of industrial model predictive control technology». In: *Control engineering practice* 11.7 (2003), pp. 733–764.

[58] Tiago P. Nascimento, António Paulo Moreira e André G. Scolari Conceição. «Multi-robot Nonlinear Model Predictive Formation Control: Moving Target and Target Absence». In: *Robot. Auton. Syst.* 61.12 (dic. 2013), pp. 1502–1515. ISSN: 0921-8890. DOI: 10.1016/j.robot.2013.07.005. URL: http://dx.doi.org/10.1016/j.robot.2013.07.005.

[59] Frank Allgower, Rolf Findeisen, Zoltan K Nagy et al. «Nonlinear model predictive control: from theory to application». In: *Journal-Chinese Institute Of Chemical Engineers* 35.3 (2004), pp. 299–316.

[60] Rien Quirynen, Milan Vukov e Moritz Diehl. «Auto generation of implicit integrators for embedded NMPC with microsecond sampling times». In: *Proceedings of the 4th IFAC nonlinear model predictive control conference* (2012), pp. 175–180.

[61] Milan Vukov et al. «Auto-generated algorithms for nonlinear model predictive control on long and on short horizons». In: *Decision and Control (CDC), 2013 IEEE 52nd Annual Conference on*. IEEE. 2013, pp. 5113–5118.

[62] Vincent Lepetit e Pascal Fua. *Monocular model-based 3D tracking of rigid objects*. Now Publishers Inc, 2005.