SAPIENZA
UNIVERSITÀ DI ROMA

Reinforcement Learning per il controllo di un quadricottero

Facoltà di Ingegneria Civile ed Industriale

Corso di Laurea Magistrale in Ingegneria Aeronautica

Candidato

Giorgio Di Liberi
Matricola 1644091

Relatore

Prof. Guido De Matteis

Correlatore

Prof. Alessandro Zavoli

Anno Accademico 2020/2021

Tesi non ancora discussa

Reinforcement Learning per il controllo di un quadricottero
Tesi di Laurea Magistrale. Sapienza – Università di Roma

Questa tesi è stata composta con LaTeX e la classe Sapthesis.

Email dell'autore: giorgio.diliberi@gmail.com

Ringraziamenti

Desidero ringraziare il mio relatore prof. Guido De Matteis sia per la fiducia, il supporto e la disponibilità che mi ha riconosciuto nello svolgimento del presente lavoro, sia per la passione trasmessa durante le sue lezioni per il mondo dell'aviazione, passione che porterò gelosamente con me nella vita professionale. Desidero poi ringraziare il mio correlatore prof. Alessandro Zavoli, senza il suo supporto tecnico non sarebbe stato possibile portare a termine questo lavoro. Desidero ringraziare di cuore la mia compagna Martina per l'amore e la pazienza con cui mi ha sostenuto durante questo percorso, senza di lei non sarei sicuramente arrivato fin qui; spero che insieme potremo raggiungere numerosi altri traguardi. Ringrazio poi mia sorella Giulia, mia madre e mio padre, se un giorno avrò una famiglia spero che sia affiatata e complice come la nostra. Ringrazio poi i miei nonni per gli insegnamenti di vita e il prezioso esempio che da sempre sanno trasmettermi.
Vorrei poi ringraziare Alessandro, per esserci sempre, per un consiglio, un conforto o per condividere un bicchiere di vino. Ringrazio poi Lorenzo e Francesca per i momenti di spensieratezza. Vorrei poi ringraziare i colleghi universitari, oggi amici, incontrati durante questo percorso, in particolare Andrea, Tiziana, Gianpaolo, Carlo, i Riccardi, i ragazzi del DBF 2017 per l'indimenticabile esperienza vissuta a Tucson e tutti gli altri colleghi che hanno contribuito a rendere questo percorso stimolante e indimenticabile.

Indice

Capitolo 1

Introduzione

1.1 Obiettivi della tesi

Lo scopo del presente lavoro di tesi è quello di studiare l'applicazione del *Reinforcement Learning* (RL) al problema del controllo di sistemi aerospaziali. In particolare lo studio è effettuato attraverso l'analisi delle prestazioni di leggi di controllo per un quadricottero di piccole dimensioni. La tecnica del RL rientra nelle metodologie di *Machine Learning* (ML) e si basa sul concetto di apprendimento per tentativi ed errori mutuato dallo studio dell'apprendimento biologico [1]. Introdurre il RL per la soluzione del problema del controllo dei velivoli consente di implementare sistemi di gestione del volo intelligenti, in grado di identificare *on-line* la dinamica dei velivoli e migliorare l'adattività della legge di controllo rispetto ai sistemi tradizionali [2]. La necessità di implementare sistemi di controllo con le caratteristiche citate è legata alla crescente complessità dei sistemi aerospaziali moderni.

L'applicazione di questa tecnica nel presente lavoro prevede l'implementazione delle funzioni di controllo attraverso approssimatori di funzione semi-lineari parametrici: le reti neurali. Tali oggetti risultano la scelta d'elezione in letteratura per implementare funzioni di controllo per sistemi MIMO [3, 4, 5, 6]. Il lavoro si articola in due fasi fondamentali: nella prima fase viene applicata la tecnica del *Reinforcement Learning* per la determinazione dei parametri delle funzioni di controllo mentre nella seconda fase si procede alla validazione della legge di controllo attraverso l'impiego in simulazione su un modello di quadricottero.

L'obiettivo finale del lavoro è realizzato attraverso l'analisi delle prestazioni di tre leggi di controllo diverse; la prima deve implementare un controllo di posizione; per la validazione di questa legge sono eseguite una serie di simulazioni in cui il modello presenta una condizione iniziale casuale, con la richiesta di raggiungere e mantenere una posizione specifica nello spazio.
Per la validazione della seconda legge, che realizza un controllo di tipo vettoriale, si imposta un controllore la cui architettura è organizzata su due livelli: l'anello esterno che elabora i riferimenti vettoriali è implementato attraverso un controllore di tipo proporzionale e l'anello interno che elabora le azioni di controllo a partire dai riferimenti vettoriali è implementato attraverso la rete neurale; il controllore

cosi realizzato viene impiegato in una missione di volo a *waypoints* nella quale viene preimpostata manualmente la rotta da seguire.
Per la validazione della terza legge, che realizza un controllo di assetto, si riportano i risultati della risposta ad un segnale di riferimento a gradino; le prestazioni della legge i controllo ottenuta sono confrontate con quelle di un regolatore di assetto di tipo proporzionale-integrale-derivativo. Il lavoro svolto è riassunto nei seguenti passi:

- **Sviluppo del modello di simulazione:** viene sviluppato un modello di simulazione del volo del qudricottero tramite la scrittura delle equazioni di Newton-Eulero;
- **Impostazione del problema di apprendimento per rinforzo:** per ognuna delle modalità di controllo da realizzare si imposta il problema di apprendimento automatico: viene definita l'interfaccia tra l'approssimatore semi-lineare e il modello al punto precedente, viene definita la funzione che calcola il *guadagno* e vengono definiti i parametri per l'applicazione del metodo di apprendimento per rinforzo alla definizione dei parametri della rete neurale;
- **validazione dei risultati:** per ogni legge di controllo ottenuta si definisce la simulazione di uno specifico piano di volo; in particolare per il controllore di posizione sono eseguite diverse simulazioni nell'ambito di una prova di tipo Monte Carlo: la condizione iniziale del modello è casuale e l'obiettivo è raggiungere e mantenere una specifica posizione; per il controllore vettoriale è impostato un piano di volo a *waypoints* ed è implementato un controllore basato su un anello interno costituito dalla legge di controllo vettoriale e un anello esterno di tipo proporzionale; per il controllore di assetto è definita una simulazione di risposta ad un segnale in ingresso a gradino: si implementa un regolatore di assetto di tipo PID per effettuare il confronto delle prestazioni dei due controllori.

1.2 Inquadramento del lavoro

1.2.1 Il quadricottero

Il quadricottero appartiene alla classe dei velivoli ad ala rotante, presenta tutti i vantaggi tipici di questi velivoli come la possibilità di decollare ed atterrare in verticale (VTOL) e la possibilità di volare a punto fisso. Inoltre consente di ridurre la complessità meccanica rispetto ad altri mezzi ad ala rotante, come gli elicotteri. Tali problematiche non sono presenti nei quadricotteri: questi sfruttano quattro rotori a passo fisso dislocati rispetto al baricentro, due di essi ruotano in verso orario e gli altri due in verso antiorario. Il controllo sulla spinta complessiva e sui momenti di beccheggio, rollio e imbardata è realizzato mediante l'opportuna variazione della velocità di ogni rotore.

Nella storia dell'aviazione i primi tentativi di realizzare dei multicotteri avvennero all'inizio del '900, quando i fratelli Breguét realizzarono il Giroplano in figura 1.1a. Il mezzo sfruttava la propulsione di un motore a combustione interna per muovere i

quattro rotori; non era presente alcun sistema di controllo, era possibile solo variare la potenza del motore e dunque la spinta complessiva. Tale caratteristica rendeva la macchina completamente incontrollabile ed era necessaria la collaborazione di operatori a terra per la stabilizzazione del mezzo.
Il velivolo Oehmichen No.2 in figura 1.1b del 1920 fu il primo quadricottero a compiere con successo delle prove di volo. Il mezzo presentava quattro rotori per la generazione della spinta di sollevamento azionati da un motore a combustione interna, in aggiunta il mezzo disponeva di 5 rotori di piccole dimensioni dislocati sui bracci del mezzo per garantire stabilità e controllabilità; tale caratteristica ha permesso a questa macchina di portare a termine diverse prove di volo durante gli anni '20 [7].
L'elicottero di Bothezat in figura 1.1c è frutto di un progetto commissionato dal governo degli Stati Uniti nel 1921. I quattro rotori erano azionati da un motore a combustione interna e presentavano pale a passo variabile per il controllo della trazione e dei momenti; la strategia di controllo cosi realizzata richiedeva al pilota un carico di lavoro molto intenso anche solo per mantenere il volo a punto fisso [8].
Il Convertawings Model A in figura 1.1d portò a termine il primo volo nel 1956; il velivolo presentava due motori a combustione interna connessi in parallelo ad una trasmissione per l'azionamento dei quattro rotori. La strategia di controllo era basata sulla presenza di un comando di passo collettivo per ognuno dei quattro rotori, analogamente al caso del velivolo di Bothezat [8]. Lo sviluppo del Curtiss-Wright VZ-7 in figura 1.1e venne completato nel 1958. I rotori erano azionati da un motore turbo albero, mentre la strategia di controllo era basata sul controllo del passo collettivo dei rotori. Per incrementare la potenza del controllo di imbardata era installato un ugello mobile alimentato dallo scarico del motore; il mezzo non riuscì tuttavia a soddisfare i requisiti in termini di carico utile ed autonomia e pertanto non entrò mai in servizio [9]. Nessuno dei velivoli citati presentava sistemi di aumento di stabilità: al pilota era richiesto di regolare singolarmente la spinta prodotta dai quattro rotori; pertanto il carico di lavoro era eccessivo anche per realizzare semplici manovre di correzione e mantenere il volo a punto fisso.

L'elenco esposto non comprende tutti i progetti che avevano come obiettivo la realizzazione di multirotori, l'interesse nella costruzione di macchine di questo tipo risiede nella loro minore complessità rispetto ad altri velivoli ad ala rotante come gli elicotteri. Questi mezzi infatti, nella configurazione tradizionale, presentano un rotore principale articolato ed un rotore di coda con pale a passo variabile; il controllo sui momenti di rollio e beccheggio e sulla trazione complessiva è realizzato variando opportunamente il passo di ogni pala del rotore principale sul giro tramite i comandi di passo ciclico e collettivo, la variazione del passo del rotore di coda controlla il momento di imbardata. Tali caratteristiche comportano complicazioni sia costruttive che di modellazione. Dal punto di vista costruttivo è necessario realizzare un collegamento tra il mozzo e le pale del rotore principale per mezzo di cerniere, affinché ogni pala abbia tre gradi di libertà rispetto al mozzo. Di conseguenza, nella fase di modellazione del mezzo finalizzata all'implementazione di leggi di controllo, è necessario formulare le equazioni del moto di ogni pala del rotore principale rispetto al mozzo in aggiunta alle equazioni del moto del corpo [11].

Grazie all'avanzamento tecnologico, il quadricottero trova ampio spazio nell'avia-

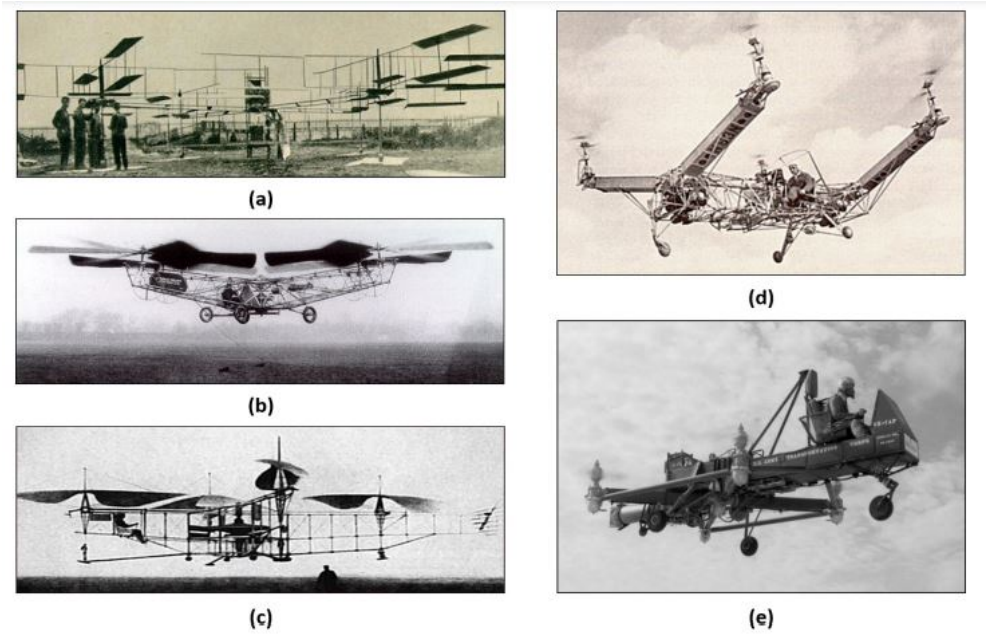

Figura 1.1. Quadricotteri con pilota: a) Breguet-Richet Gyroplane, b) Oehmichen No.2, c) Bothezat helicopter, d) Convertawings Model A, e) Curtiss-Wright VZ-7 [10].

zione moderna come *Unmanned Aerial Vehicle* (UAV) ovvero velivolo senza pilota; sono disponibili diverse tipologie di mezzi che si distinguono per disposizione dei rotori, dimensioni e strategie di controllo [12]. Particolare impulso al settore è stato dato dalla recente disponibilità in commercio di sensori ed attuatori Micro-elettro-meccanici (MEMS) leggeri ed economici, che consentono lo sviluppo di sistemi di controllo automatico digitali. Inoltre la disponibilità di accumulatori elettrici ad alta densità di energia e di motori elettrici con controllo elettronico della velocità di rotazione consente di sfruttare rotori a passo fisso, riducendo ulteriormente la complessità costruttiva rispetto alle più grandi e pesanti macchine *manned* citate in precedenza.

I campi di utilizzo dei quadricotteri di piccole dimensioni sono molteplici, a seconda della tipologia di *payload* per cui vengono realizzati.
Uno degli impieghi di maggiore diffusione in ambito civile è quello della sorveglianza ed esplorazione aerea, dove il quadricottero costituisce una piattaforma capace di trasmettere immagini ad una stazione di terra; in questo modo si compiono ricognizioni in scenari dove è rischioso o costoso inviare operatori umani. Un caso particolare di questo utilizzo è la ricerca e soccorso di dispersi in mare. Recentemente infatti un gruppo di ricerca della *Royal Navy* ha testato con successo il quadricottero Minerva T-150; il mezzo ha lo scopo di localizzare autonomamente i dispersi e comunicare la posizione esatta alla squadra di soccorso, alleggerendone il carico di lavoro e limitando i rischi. Per realizzare questi obiettivi il mezzo è equipaggiato con dei sistemi di controllo che gli permettono di volare in modo completamente autonomo, oltre che con delle unità di calcolo per processare le immagini e riconoscere autonomamente oggetti e persone [13].

Figura 1.2. Ascending Technologies Hummingbird: esempio di quadricottero moderno.

Un altro campo di utilizzo di interesse riguarda l'implementazione di reti robotiche di sensori *wireless*, ovvero di reti locali per lo scambio di dati basate su nodi robotici. In questo contesto il quadricottero svolge la funzione di un nodo della rete e costituisce la piattaforma su cui installare sensori per il monitoraggio ambientale; ogni mezzo appartenente alla rete deve implementare anche i necessari dispositivi per comunicare con gli altri nodi. Questo consente di sviluppare un sistema di osservazione distribuito su diverse macchine ed in grado di operare in ambienti privi di infrastrutture di telecomunicazione [14].

I campi di utilizzo citati richiedono l'implementazione di sistemi di controllo complessi, organizzati in architetture gerarchiche; tali sistemi devono sia garantire la stabilità del mezzo che renderlo autonomo nella navigazione, compensando anche azioni di disturbo non modellizzabili in fase di progettazione, ad esempio guasti o malfunzionamenti dei componenti, turbolenze o perturbazioni atmosferiche casuali. Data l'ampia diffusione in ambito civile, i quadricotteri e le operazioni in cui sono impiegati trovano oggi inquadramento nella normativa aeronautica EASA con i regolamenti 2019/945 e 2019/947. In particolare il modello di quadricottero su cui è basato il presente lavoro afferisce alla categoria di massa C1 del regolamento: mezzi con massa massima al decollo (MTOM) inferiore a $0.9\,kg$ [15]. In tabella 1.1 sono riassunte le categorie del regolamento.

1.2.2 Strategie di controllo

Il problema del controllo automatico applicato ai quadricotteri è affrontato da diverse decadi: sono disponibili numerosi lavori che propongono l'implementazione di sistemi di controllo di varia complessità per raggiungere diversi obiettivi. I sistemi di controllo allo stato dell'arte per quadricotteri di piccole dimensioni si dividono in tre categorie fondamentali: controllori lineari [16, 17], controllori non lineari [18] e controllori basati su apprendimento automatico [19, 10].

Categoria	MTOM [kg]
C0	0.25
C1	0.9
C2	4
C3	25

Tabella 1.1. Categorie e massa massima al decollo secondo la normativa EASA per gli UAV [15].

I regolatori lineari costituiscono il primo approccio storicamente adottato nella progettazione di sistemi di controllo automatico per i velivoli, pertanto attualmente sono la categoria in assoluto più rappresentata sia in ambito di ricerca che industriale. Si citano in particolare due tipologie di regolatori appartenenti a questa categoria: PID ed LQR. Il regolatore proporzionale-integrale-derivativo è il più diffuso, controllori basati su leggi di questo tipo sono generalmente organizzati su logiche gerarchiche basate su più anelli a cascata; tale caratteristica consente di dividere lo sviluppo de sistema in relazione all'obiettivo che ogni *loop* deve raggiungere: tipicamente l'anello di controllo interno è preposto all'aumento di stabilità e al controllo diretto del velivolo mentre l'anello esterno ha lo scopo di implementare modalità di volo complesse come l'inseguimento di traiettoria o il controllo di posizione, generando i riferimenti per l'anello interno. Diverse strade possono essere intraprese per la definizione dei parametri di questi controllori, la più semplice è quella empirica, in cui il valore dei parametri è determinato manualmente tramite dei tentativi di impiego diretto del controllore. Dato l'elevato costo e la poca efficienza di questo approccio, sono stati proposti metodi per la determinazione dei parametri che si basano sul modello di simulazione del velivolo da controllare e fanno riferimento alle funzioni di trasferimento del sistema completo [20, 16].

I regolatori LQR rappresentano una soluzione al problema del controllo ottimo [19, 18]: i parametri del controllore sono determinati come i punti di minimo di una funzione di costo che descrive le prestazioni del sistema controllato. Tali metodi presentano un problema di notevole importanza: le prestazioni dipendono dall'accuratezza del modello di simulazione su cui sono costruiti. Inoltre i regolatori LQR fanno riferimento a sistemi lineari, mentre il comportamento dinamico dei velivoli è caratterizzato da azioni aerodinamiche con relazioni non lineari rispetto alle variabili di stato. Per assecondare le esigenze della progettazione di questi controllori si procede dunque alla linearizzazione del modello di simulazione in una condizione di volo di equilibrio; tale operazione comporta che la validità dei risultati ottenuti in termini di prestazioni del sistema di controllo sia ridotta ad una porzione ristretta dell'inviluppo di volo. Per risolvere tale problematica è formulata la tecnica del *Gain scheduling* che permette di realizzare dei sistemi di controllo *adattivi* basati su leggi lineari. L'applicazione di tale tecnica è ormai consolidata nell'industria aeronautica, tuttavia risulta molto laboriosa in quanto prevede che il processo di determinazione dei parametri delle leggi di controllo venga effettuato linearizzando il

modello di simulazione in diverse condizioni di equilibrio; in seguito, in fase di impiego del sistema, il valore dei parametri è selezionato in base alla condizione di volo [21].

Le tecniche di controllo basate su leggi non lineari sono sviluppate al fine di superare alcune limitazioni dei controllori lineari, in particolare quella legata alla ridotta validità delle prestazioni ottenute da controllori basati su modelli linearizzati del sistema controllato [19, 22]. Queste tecniche hanno dunque come obiettivo principale l'estensione dell'inviluppo di volo in cui il controllore è efficace. Un altro obiettivo dei controllori non lineare è realizzare manovre complesse: i quadricotteri presentano infatti sei gradi di libertà e dispongono di quattro ingressi di controllo corrispondenti alla trazione generata dai rotori. Tale aspetto comporta un vincolo importante da tenere in considerazione nella definizione del paradigma di controllo e rende impraticabile il compimento di acrobazie per mezzo di controllori lineari.

1.2.3 Strategie di controllo basate sul metodo di apprendimento automatico

Il metodo oggetto del presente lavoro appartiene alla categoria dei controllori basati su apprendimento automatico. In particolare l'apprendimento per rinforzo (*Reinforcement Learning*) consiste nell'applicazione del concetto di apprendimento per tentativi ed errori, mutuato dallo studio del processo decisionale messo in atto dagli organismi biologici evoluti [1, 2]. Questa tecnica prevede che la legge di controllo sia implementata attraverso una funzione parametrica e nel presente lavoro sono utilizzate le reti neurali come strumenti matematici per approssimare l'andamento di funzioni non lineari; la rete neurale riceve come ingresso un'informazione sullo stato del mezzo e restituisce in uscita le azioni di controllo calcolate.

Il metodo di apprendimento per rinforzo è applicato per la determinazione dei parametri della funzione di controllo: in letteratura sono disponibili numerosi algoritmi che formulano metodi matematici di applicazione della tecnica [2, 23, 4, 24, 6]. Si parla di apprendimento automatico o allenamento della rete neurale in quanto l'algoritmo determina l'aggiornamento dei parametri della funzione di controllo in base ai risultati ottenuti applicando direttamente quest'ultima sul sistema da controllare. Tale aspetto costituisce un'analogia con ciò che accade, ad esempio, nel cervello di un essere umano che apprende un movimento fisico: si compiono dei tentativi iniziali cui consegue il fallimento; in seguito, grazie all'esperienza accumulata, il controllo neuro-muscolare si aggiorna fino ad ottenere un risultato accettabile.

Un vantaggio di questi metodi rispetto a quelli convenzionali è rappresentato dal fatto che la definizione della legge di controllo non richiede a priori alcuna informazione sul sistema da controllare; i parametri della funzione sono determinati automaticamente in base all'esperienza accumulata nel controllo del sistema. In questo senso si parla di tecniche di progettazione dei controllori *model-free* [1], in contrapposizione alle tecniche *model-based* afferenti alle categorie alla sezione precedente [19, 2]. Questo aspetto è determinante nel contesto del controllo di sistemi MIMO come i quadricotteri per i quali è complesso formulare una modellazione che

tenga conto in modo accurato di tutti gli effetti dinamici che li caratterizzano.

Una conseguenza importante di questo aspetto è che il metodo proposto è applicabile per implementare funzioni di controllo a diversi livelli. Applicato alla realizzazione di controllori di basso livello per la regolazione di assetto il RL consente di migliorare l'efficienza del controllore rispetto alle tecniche canoniche; soprattutto in presenza di incertezze sulla modellazione dinamica del mezzo o di azioni di disturbo al processo non modellizzabili, ad esempio, turbolenze aerodinamiche di natura casuale. Applicato invece alla realizzazione di funzioni di alto livello consente di ottenere sistemi dotati di autonomia decisionale che operino in condizioni non previste durante la progettazione; in questo contesto rientrano le funzioni di generazione della traiettoria di riferimento per sonde di esplorazione spaziale profonda [2].

Il presente lavoro propone l'applicazione della tecnica per ottenere tre funzioni di controllo di basso livello: i segnali di riferimento in ingresso sono per il primo controllore la posizione, per il secondo la velocità e la direzione e per il terzo l'assetto di riferimento; in tutti i casi elencati, la funzione di controllo è implementata da una rete neurale che restituisce in uscita le azioni in termini di trazione complessiva e momenti in assi corpo. L'obiettivo della prima legge di controllo è dimostrare che il RL consente di ottenere funzioni in grado di regolare l'assetto del quadricottero a partire da qualsiasi condizione iniziale per raggiungere una posizione di riferimento. Lo scopo della seconda legge di controllo è provare che un controllore ottenuto con questo metodo opera efficacemente in un'architettura a cascata di cui costituisce l'anello di controllo interno. Obiettivo della terza funzione di controllo è infine quello di dimostrare che la legge per il controllo di assetto ottenute con questo metodo presenta prestazioni superiori rispetto ad un regolatore di assetto di tipo PID.

Negli ultimi anni sono stati pubblicati diversi lavori in cui si propone l'applicazione del RL per la realizzazione di funzioni di controllo di UAV ad ala rotante di piccole dimensioni, si riportano alcuni di quelli i cui obiettivi sono analoghi a quelli del presente lavoro. In [3] Abeell et al. propongono l'applicazione del *Reinforcement learning* per ottenere un controllore in grado di realizzare manovre acrobatiche con un elicottero di piccole dimensioni. I parametri della funzione di controllo sono determinati sulla base di dati raccolti in voli realizzati da un pilota umano; i ricercatori concludono che solo questo tipo di tecniche consentono l'implementazione di controllori automatici in grado di realizzare manovre aggressive o acrobatiche.

In [23] Waslander et al. sfruttano la tecnica del RL per ottenre un controllore di quota per un quadricottero. I ricercatori inseriscono nel modello dinamico dei disturbi come accelerazioni casuali con distribuzione Gaussiana allo scopo di introdurre nel sistema un rumore non modellizzabile; inoltre viene presentato un confronto delle prestazioni con un controllore non lineare di tipo *sliding mode*.

Nel lavoro [4] si propone un algoritmo basato sul RL per realizzare un controllore di un quadricottero basato su una rete neurale; la rete riceve come ingresso lo stato completo del mezzo e restituisce in uscita i comandi sulla trazione complessiva e sui momenti in assi corpo. Scopo della ricerca è ottenere una funzione di controllo per

posizionare il quadricottero in un punto stabilito a partire da una condizione casuale e in presenza di disturbi; i risultati sono validati con dei test sperimentali.

In [24] i ricercatori sfruttano diversi algoritmi per determinare i parametri di reti neurali preposte al controllo di assetto di un quadricottero; vengono forniti dei riferimenti in termini di angoli e velocità di rotazione e la funzione restituisce il valori dei momenti da applicare. Le prestazioni dei controllori ottenuti vengono confrontate con quelle di regolatori PID: le reti neurali risultano più efficaci sia nel transitorio che nella gestione di disturbi.

In [6] è proposta la realizzazione, tramite algoritmi di RL, di controllori che implementano due modalità di volo di un quadricottero: posizionamento a partire da una condizione casuale e inseguimento di riferimenti vettoriali. L'obiettivo è quello di ottenere dei controllori efficaci in tutto l'inviluppo di volo del quadricottero. I ricercatori dimostrano che in condizioni di volo reale un metodo per migliorare le prestazioni a stazionario del controllore basato su rete neurale è interfacciarlo con regolatori proporzionali-integrali.

1.3 Organizzazione della Tesi

L'elaborato è organizzato come segue: nel secondo capitolo si espongono alcuni elementi chiave dell'apprendimento automatico per rinforzo, dell'algoritmo e della libreria *software* usata nel lavoro. Nel terzo capitolo vengono esposte le equazioni del moto formulate per il quadricottero; vengono evidenziati gli elementi chiave della modellazione dinamica e dell'integrazione numerica del modello di simulazione. Il capitolo quarto tratta dell'impostazione e del risultato dell'allenamento delle *policies* per la realizzazione dei tre modi di volo. Il quinto capitolo tratta i risultati delle simulazioni di validazione delle tre *policies* una volta terminato l'allenamento; limitatamente al controllo di assetto si propone un confronto delle prestazioni del controllore allenato con un controllore lineare PID. Infine il sesto capitolo riporta le conclusioni del lavoro svolto e alcuni spunti per futuri lavori sul tema.

Capitolo 2

Reinforcement Learning

Il *Reinforcement Learning* (RL) è una delle tre branche del *Machine Learning* (ML) [1], le altre due sono il *Sustained Learning*(SL) [25] e il *Unsustained Learning* (UL) [25]; il concetto alla base di questa tecnica è addestrare in modo automatico un agente a controllare un ambiente conoscendone solo lo stato attuale al fine di massimizzare una funzione scalare denominata *reward* o guadagno.
Le modalità con cui l'agente controlla l'ambiente dipendono dalla tipologia di ambiente da controllare e dalle sue caratteristiche, oltre che dalla tipologia di interfaccia che collega l'agente con l'ambiente. Nel presente lavoro l'ambiente da controllare è il modello di quadricottero mentre l'agente è l'insieme degli elementi che elaborano ed attuano il controllo, intervenendo sul valore della trazione generata da ogni motore. Ad ogni passo temporale l'agente riceve in ingresso lo stato cinematico del modello ed elabora le azioni di controllo.

Ad ogni possibile stato del modello è associato un guadagno scalare: la formulazione di questo valore è scelta dal programmatore in fase di impostazione del problema e dipende dall'obiettivo particolare che si vuole raggiungere. Se, ad esempio, si imposta l'apprendimento per ottenere una funzione di controllo di assetto del quadricottero, il guadagno deve essere inversamente proporzionale all'errore sui riferimenti in termini di angoli di assetto. L'agente riceve ad ogni istante l'errore di assetto del mezzo rispetto ai riferimenti ed il valore del guadagno ad essi associato elaborando l'azione di controllo. In seguito, sulla base del guadagno ottenuto in una serie di passi temporali, l'agente elabora l'aggiornamento della funzione che mappa lo spazio degli stati nello spazio delle azioni: si fa riferimento a questo processo con la locuzione *on-line learning*. L'aggiornamento è rivolto ad ottenere la funzione di controllo che massimizza il guadagno ad ogni passo temporale.
La natura dello spazio degli stati e dello spazio delle azioni dipende dal particolare problema in esame: nel problema oggetto del lavoro gli stati dell'ambiente coincidono con le variabili di stato del modello di quadricottero mentre le azioni determinate dall'agente sono le variabili di controllo del mezzo; ertanto sia lo spazio degli stati che quello delle azioni appartengono all'insieme dei numeri reali.

La tecnica in esame si differenzia dal SL in relazione al concetto di apprendimento *on-line* [25]. Nel SL è prevista la disponibilità a priori di un set di dati contenente

per ogni valore degli stati possibile l'azione corretta da intraprendere; nel RL non è invece noto a priori quale sia l'azione corretta da intraprendere dato un certo stato dell'ambiente, questa è determinata tramite l'interazione diretta dell'agente sull'ambiente in base al valore del guadagno. Tale caratteristica rende il SL inadatto all'applicazione proposta in questo lavoro data la complessità dello spazio degli stati e dello spazio delle azioni che caratterizzano il modello di quadricottero: è impossibile valutare per ogni possibile stato cinematico la combinazione di azioni corretta in base all'obiettivo della funzione di controllo da realizzare.
La tecnica del UL prevede che l'agente riceva in ingresso lo stato dell'ambiente per un certo numero di passi temporali [25]; in seguito l'agente riconosce gli schemi nello stato attuale dell'ambiente al fine di prevederne lo stato futuro; questa caratteristica distingue la tecnica del UL dal RL dove invece l'agente, dato lo stato dell'ambiente, fornisce in uscita un'azione di controllo.

Le radici storiche di questa tecnica risiedono nell'incontro di due particolari campi di ricerca. Il primo è quello della psicologia, con riferimento al concetto di apprendimento per tentativi ed errori introdotto nello studio del comportamento animale nel XIX secolo dallo studioso A. Bain [25]. Il secondo campo di ricerca è quello dei controlli automatici in riferimento al problema del controllo ottimo di sistemi dinamici; uno dei primi lavori su questo tema venne proposto negli anni '50 da R. Bellman [25]. Lo studio del controllo ottimo ha come obiettivo l'implementazione di funzioni di controllo che minimizzano una funzione di costo descrittiva delle prestazioni del sistema controllato.
In particolare il metodo del *Reinforcement Learning* consiste nell'elaborare i parametri della funzione di controllo in base ai risultati ottenuti da quest'ultima applicata direttamente al sistema da controllare con l'obiettivo di massimizzare il guadagno ottenuto ad ogni passo temporale; in questo modo si riproduce il processo di apprendimento biologico attraverso l'applicazione di un algoritmo matematico [1].

2.1 Elementi del sistema RL

Si distinguono quattro elementi principali:

- la *Policy* è la relazione funzionale che intercorre tra gli stati e le azioni; a seconda della natura del problema specifico assume la forma di una tabella contenente le probabilità di intraprendere determinate azioni dati determinati valori dello stato; alternativamente è un approssimatore funzionale non lineare parametrico (es. rete neurale) che mappa lo spazio degli stati nello spazio delle azioni; la *policy* è l'elemento del sistema che viene aggiornato durante l'allenamento;

- il *guadagno* è un valore scalare dipendente dallo stato ce viene calcolato ad ogni passo temporale; è formulato a partire da una relazione per cui a ogni possibile stato corrisponde un solo valore del *guadagno*; l'obiettivo dell'agente è intraprendere l'azione che massimizza questo valore; nel caso in esame nel presente lavoro il *guadagno* è calcolato da una funzione in base al valore delle variabili di stato del quadricottero; la formulazione dipende dal particolare

problema che si intende risolvere: ad esempio nel caso del controllore di posizione, la funzione che calcola il guadagno è inversamente proporzionale all'errore di posizione del mezzo e alla sua velocità; in questo modo si ottiene il massimo valore del *guadagno* quando il mezzo si trova nella posizione di riferimento con velocità nulla;

- il *valore* di un'azione rappresenta la somma dei guadagni attesi nei passi temporali futuri se si intraprende l'azione in esame al passo temporale attuale; il valore del guadagno stimato per i passi temporali futuri è moltiplicato per un parametro reale $\gamma \in [0, 1]$ denominato fattore di sconto; a differenza del guadagno il valore ha un significato di lungo periodo e rappresenta un elemento da tenere in considerazione durante l'allenamento; la classe di algoritmi che sfrutta proprio il *valore* per eseguire gli aggiornamenti della *policy* è denominata *Q-learning* [26];

- il *modello* dell'ambiente rappresenta un elemento opzionale che permette all'agente di effettuare delle previsioni sul comportamento futuro se si intraprende una determinata azione, restituendo una serie di scenari possibili in termini di *valore*; la presenza di un modello a disposizione dell'agente classifica un algoritmo come *model-based* al contrario si hanno algoritmi di tipo *model-free* come quello adottato nel presente lavoro.

L'agente è l'entità che ricava lo stato dell'ambiente, elabora ed applica le azioni; relativamente al controllo del modello di quadricottero l'agente è la funzione che elabora l'azione di controllo a partire dallo stato del mezzo; nel caso di un quadricottero reale l'agente è l'insieme delle apparecchiature elettroniche che calcolano lo stato del mezzo, l'unità di calcolo che elabora le azioni di controllo e il sistema di attuazione costituito dai rotori.

2.2 Descrizione degli elementi usati nel lavoro

Nel presente lavoro lo scopo è ottenere delle funzioni di controllo per mappare lo spazio continuo degli stati di un quadricottero nello spazio continuo delle azioni di controllo; la relazione funzionale che realizza la mappatura è costituita da una rete neurale artificiale; l'algoritmo utilizzato per gli allenamenti è *Proximal Policy Optimization* (PPO) appartenente alla classe dei *Policy gradient method* in quanto l'aggiornamento della *policy* viene valutato sulla base del gradiente della funzione di *valore* rispetto ai parametri della *policy*. La formulazione matematica dell'algoritmo è presentata nel lavoro [27].

Come anticipato, nei problemi affrontati nel presente lavoro sia lo spazio degli stati che lo spazio delle azioni sono di natura continua; per ulteriori dettagli sulla definizione di questi spazi relativamente alla risoluzione di ognuno dei tre problemi di controllo affrontati si rimanda al capitolo 4. Tale algoritmo è indicato per affrontare problemi in cui lo spazio degli stati e quello delle azioni sono continui.
Nei problemi in cui invece lo spazio di stati e quello delle azioni hanno natura discreta, la *policy* è una *look-up table* che per ogni stato-azione riporta una probabilità basata

sul valore; l'azione selezionata dall'agente ad ogni passo temporale è quella che presenta la probabilità maggiore dato lo stato attuale. Per questa tipologia di problemi, l'applicazione del metodo valuta l'aggiornamento della *policy* determinando i valori delle probabilità associate ad ogni coppia stato-azione.

L'algoritmo PPO permette di allenare l'agente collezionando esperienza diretta e non richiede l'utilizzo di un *modello* per prevedere il comportamento dell'ambiente; a tal proposito si precisa che il termine *modello* in questo contesto non è relazionato al fatto che gli allenamenti sono stati effettuati impiegando un modello simulato del quadricottero, ma fa riferimento al fatto che l'unico dato di cui dispone l'agente per elaborare le azioni è lo stato del quadricottero.

2.2.1 Reti neurali artificiali

Le reti neurali artificiali sono approssimatori di funzioni non lineari; il loro utilizzo nei sistemi di apprendimento per rinforzo è ampiamente esplorato per diverse applicazioni: dal riconoscimento di immagini al controllo diretto di *robot*; l'applicazione di algoritmi di RL ad agenti costituiti da reti neurali rappresenta la riproduzione artificiale del processo di apprendimento biologico [1].

Nel presente lavoro si utilizzano reti di tipo *Multi-Layer Perceptron* in cui ogni nodo della rete è connesso ad ogni nodo degli strati adiacenti e a ciascuna connessione è associato un peso reale. Si parla di approssimatori semi-lineari in quanto l'ingresso di ogni nodo è valutato come combinazione lineare dell'uscita dei nodi dello strato precedente a cui viene poi applicata una funzione non lineare denominata funzione di *attivazione*; nel presente lavoro è stata scelta la funzione *tangente iperbolica* perchè presenta dominio compreso nell'intervallo $(-\inf, +\inf)$ e codominio compreso nell'intervallo $[-1, 1]$; tale caratteristica è fondamentale per l'applicazione della funzione di controllo al problema in esame in quanto i valori in uscita dalla rete neurale costituiscono le azioni di controllo per il quadricottero che assumono valori sia negativi che positivi. L'uscita del nodo i-esimo situato nello strato j-esimo si calcola come:

$$Y_{i,j} = \tanh\left(X_{i,j}\right) \quad con \quad X_{i,j} = \boldsymbol{w}^{i,j-1} \cdot \boldsymbol{Y}^{i,j-1} + b_{i,j} \tag{2.1}$$

dove $Y_{i,j}$ è il valore scalare di uscita dal nodo in esame, $X_{i,j}$ è il valore in ingresso allo stesso nodo, calcolato come prodotto scalare tra il vettore dei pesi $\boldsymbol{w}^{i,j-1}$ e il vettore delle uscite $\boldsymbol{Y}^{i,j-1}$ dei nodi dello strato $j-1$; infine viene aggiunto un termine di *bias* $b_{i,j}$. Durante l'allenamento, partendo dai dati raccolti impiegando la rete sull'ambiente, vengono modificati i valori dei pesi $w_k^{i,j-1}$ e dei *bias* $b_{i,j}$ associati ad ogni connessione al fine di ottenere la migliore mappatura tra stati e azioni. Una descrizione schematica di una rete neurale è mostrata in figura 2.1.

L'architettura delle reti usate in tutti gli agenti allenati presenta uno strato di ingresso, uno strato di uscita e due strati intermedi denominati *hidden layers*. Il numero di nodi che compone gli strati intermedi è 64 mentre il numero di nodi di ingresso e uscita dipende dalla particolare funzione di controllo che si vuole ottenere; il numero di nodi degli strati intermedi determina la *profondità* della rete e va

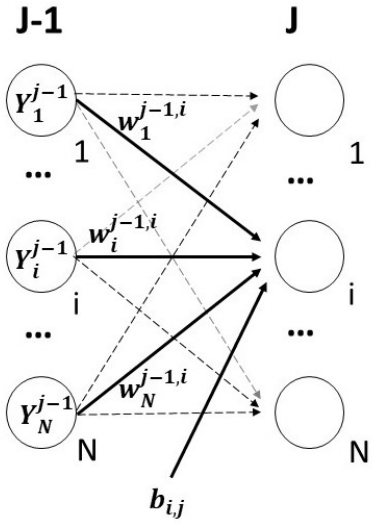

Figura 2.1. Schema di una rete neurale MLP a due strati

impostato tenendo conto del problema del *fitting*.

Un approssimatore di funzione con un numero di nodi intermedi eccessivamente ridotto si trova nella condizione di *under-fitting*: tale condizione comporta che la rete neurale ottenuta non ha un numero sufficiente di connessioni per mappare ogni combinazione degli ingressi in un valore delle azioni efficace. Nel caso in esame ciò che si ottiene in questa condizione è un controllore che presenta una ridotta efficacia in alcune regioni dell'inviluppo di volo.
Al contrario l'*over-fitting* è la condizione in cui l'approssimatore presenta un numero di nodi eccessivo in relazione all'applicazione in esame. Tale condizione non comporta necessariamente che il controllore ottenuto sia inefficace; tuttavia, dato che i parametri della rete sono determinati con un processo di apprendimento conseguente all'impiego diretto della stessa sull'ambiente, in presenza di *over-fitting* si incorre in una perdita di generalità del controllore ottenuto ovvero ad una sua ottimizzazione associata specificamente alle combinazioni di stati incontrate nel processo di apprendimento [28].

Nella scelta del numero di nodi intermedi va tenuto conto che il costo in termini temporali per la determinazione dei parametri è direttamente proporzionale alla *profondità* della rete: il numero finale è dunque una scelta di compromesso tra *budget* temporale a disposizione e problemi di *fitting*. Nel presente lavoro la scelta è stata fatta su base empirica.

La figura 2.2 rappresenta l'architettura delle reti usate nei controllori sviluppati in questo lavoro.

Infine la figura 2.3 riporta gli elementi del sistema di apprendimento automatico:

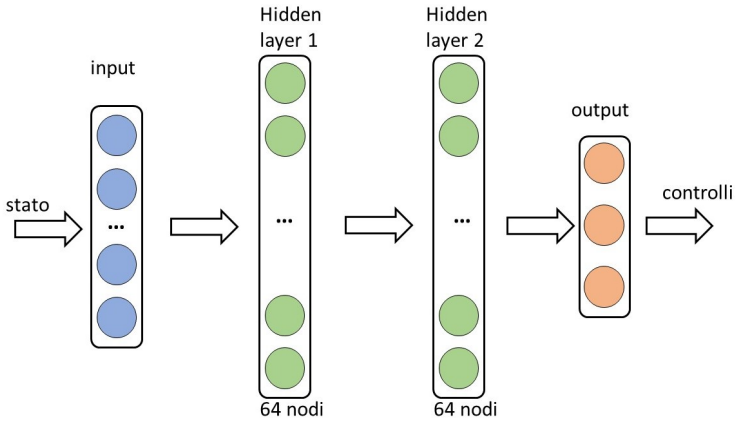

Figura 2.2. Architettura delle reti MLP 64x64.

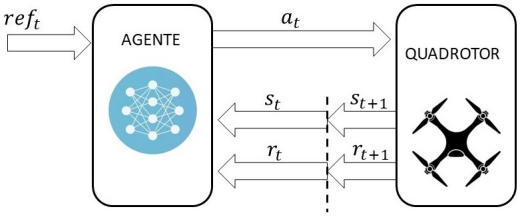

Figura 2.3. Architettura del sistema di apprendimento automatico.

l'agente riceve lo stato del *quadrotor* s_t, il *guadagno* r_t e i riferimenti ref_t al passo temporale t ed elabora l'azione a_t; durante le sessioni di allenamento sono valutate le modifiche da apportare alla *policy*. Il procedimento adottato nel presente lavoro si divide in due fasi: in primo luogo è applicato il processo di determinazione dei parametri della *policy* attraverso l'algoritmo; in questo stadio la funzione di controllo interviene sul modello di quadricottero al solo scopo di ottenere i dati necessari all'aggiornamento dei parametri; in seguito la funzione di controllo ottenuta viene impiegata sul modello di quadricottero in simulazioni di volo atte a verificare le prestazioni del sistema.

Si sottolinea che l'applicazione della tecnica non richiede obbligatoriamente una netta separazione tra la definizione dei parametri e l'impiego della funzione di controllo. In molti casi infatti il metodo consente di mantenere attivo il processo di aggiornamento della *policy* anche durante l'impiego della funzione di controllo. Tale caratteristica in un sistema reale consente di garantire la robustezza del controllore durante la vita operativa grazie all'adattamento della funzione ai cambiamenti del velivolo legati all'usura delle componenti.

2.2.2 Algoritmo *Proximal Policy Optimization*

Il PPO appartiene alla classe dei metodi del gradiente *attore-critico*; per sfruttare questo algoritmo l'agente deve implementare una funzione parametrica che costituisce la mappatura stato-azione denominata *attore* e una funzione parametrica che stima il *valore* di ogni coppia stato-azione denominata *critico*; l'attore si rappresenta come $\pi(a_t|s_t, \boldsymbol{\theta}) = \pi_{\boldsymbol{\theta}}$ dove a_t è l'azione all'istante t elaborata conoscendo lo stato s_t con vettore dei parametri della *policy* corrispondente a $\boldsymbol{\theta}$ [27, 29].
La relazione matematica alla base dei metodi *Policy-gradient* è:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha \vec{\bigtriangledown}_\theta J(\pi_\theta) \tag{2.2}$$

dove $\boldsymbol{\theta}_{t+1}$ è il vettore dei parametri aggiornato, $\boldsymbol{\theta}_t$ è il vettore dei parametri non aggiornato, α è un parametro reale e $\vec{\bigtriangledown}_\theta J(\pi_\theta)$ rappresenta il gradiente della funzione J rispetto ai parametri non aggiornati. La funzione J descrive le prestazioni della *policy*, ovvero il guadagno raccolto in un episodio dall'agente con valore dei parametri $\boldsymbol{\theta}_t$. La relazione 2.2 indica che l'aggiornamento dei parametri è eseguito nella direzione di massima crescita della funzione J che si scrive come:

$$J(\pi_\theta) = E[G(\tau_\theta)] \quad con \quad G(\tau_\theta) = \sum_{t=0}^{T} \gamma^t \, r_t \tag{2.3}$$

dove τ_θ è una specifica traiettoria descritta dall'ambiente sotto la *policy* π_θ in un episodio, $G(\tau_\theta)$ è il *ritorno* ovvero la somma dei *guadagni* ottenuti seguendo la traiettoria τ_θ, scontati di un fattore reale $\gamma \in [0, 1]$ che consente di modulare il peso dei *guadagni* futuri. La traiettoria è:

$$\tau_\theta = (s_0, a_0, s_1, a_1, ..., s_t, a_t, ..., s_{T-1}, a_{T-1}, s_T) \tag{2.4}$$

dove s_t è lo stato al passo temporale t, cui consegue l'azione a_t che a sua volta è seguita dallo stato s_{t+1}; l'episodio termina nello stato s_T dunque la traiettoria descritta è caratterizzata da un orizzonte temporale finito pari a T passi.

Teorema *Policy Gradient*

Il teorema *Policy Gradient* risolve il problema del calcolo di $\bigtriangledown_\theta J(\pi_\theta)$ che è altrimenti impossibile. Il teorema dimostra che [30]:

$$\bigtriangledown_\theta \, J(\pi_\theta) = E\left[\sum_{t=0}^{T} \bigtriangledown_\theta log(\pi_\theta(a_t|s_t)) \cdot G(\tau_\theta)\right] \tag{2.5}$$

dove l'espressione della *policy* $\pi_\theta(a_t|s_t)$ indica la probabilità di compiere a_t dato s_t, mentre il termine $G(\tau_\theta)$ è il ritorno nell'episodio ottenuto a partire dallo stato s_0 applicando la *policy* caratterizzata dal vettore dei parametri $\boldsymbol{\theta}$. Sia il termine $log(\pi_\theta(a_t|s_t))$ che il termine $G(\tau_\theta)$ nella 2.5 sono noti a valle di uno o più episodi in cui l'agente interagisce con l'ambiente e, contemporaneamente, il calcolo del gradiente è effettuato numericamente a valle di un numero arbitrario di episodi.

Algoritmo PPO

Caratteristica peculiare del PPO è limitare il grado di variazione dei parametri per non degradare eccessivamente le prestazioni dell'agente. Questo fenomeno di verifica se il gradiente valutato numericamente conduce la *policy* in una regione in cui la mappatura stato-azione comporta ritorni bassi; ciò implica che i nuovi dati raccolti *on-line* per l'aggiornamento hanno scarsa qualità, innescando un circolo vizioso che pregiudica il risultato finale dell'allenamento. In primo luogo viene valutata la funzione *obiettivo* che vale:

$$L^{PG} := \hat{E}_t \left[log\pi(a_t|s_t) \cdot \hat{A}_t \right] \tag{2.6}$$

corrispondente al valore atteso della probabilità di avere l'azione a_t dato lo stato s_t moltiplicato per il valore relativo dell'azione denominato *vantaggio* $\hat{A}_t$. Il termine $\hat{A}_t$ rappresenta il valore relativo di a_t e si calcola come:

$$\hat{A}_t = G_t - v(s) \tag{2.7}$$

dove G_t è il ritorno al passo temporale t pari a:

$$\sum_{k=0}^{T} \gamma^k \, r_{t+k-1} \tag{2.8}$$

mentre $v(s)$ è la stima del valore dello stato s_t effettuata dalla rete neurale che svolge la funzione di *critico*; il calcolo di A_t viene effettuato a valle della conclusione di un numero prefissato di *episodi*. La rete neurale che svolge la funzione di *critico* viene aggiornata costantemente durante l'allenamento a valle di ogni episodio basandosi sulla differenza tra il *valore* stimato e quello effettivamente raccolto richiamando la tecnica di *apprendimento supervisionato* (SL).
La funzione *obiettivo* L^{PG} viene ottimizzata dall'algoritmo tramite la seguente espressione:

$$L^{CLIP} = \hat{E}_t \left[min(r_t(\boldsymbol{\theta})\hat{A}_t, clip(r_t(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon)\hat{A}_t) \right] \tag{2.9}$$

dove:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta old}(a_t|s_t)} \tag{2.10}$$

rappresenta il rapporto tra la probabilità di compiere a_t dato lo stato s_t sotto la *policy* aggiornata; il termine $clip(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)$ nella 2.9 indica che viene determinato il minimo tra la funzione precedente ed un intorno definito dal parametro ϵ che tipicamente assume un valore pari a 0.2. L'*obiettivo* ha lo scopo di aggiornare la *policy* per rendere più probabili mappature stato-azione con *ritorno* maggiore, imponendo una limitazione alla modifica della *policy*. Inoltre, dal momento che il valore di A_t è una stima rumorosa del valore relativo di un'azione, si effettua una media su una serie di episodi diversi; tale insieme viene denominato *batch*.
Si riporta di seguito lo pseudo codice dell'algoritmo:

Il flusso indica che la raccolta dei dati su cui valutare l'aggiornamento dei pesi viene effettuato impiegando N *agenti* in parallelo per un numero di passi temporali

Algorithm 1 Pseudo-codice dell'algoritmo PPO in un sistema *attore-critico*

```
1: procedure PPO
2:     for iteration = 1, 2, ... do
3:         for actor = 1, 2, ... do
4:             impiego policy π_θold in ambiente per T steps
5:             stima del vantaggio Â_t ∀t ∈ [0, T]
6:         ottimizzazione di L tramite θ, per K epoche
7:         θ_old ← θ
```

pari a T, la stima della funzione $\hat{A}$ viene fatta in ogni passo tramite il ritorno raccolto; in seguito viene eseguita l'ottimizzazione di L con riferimento al vettore dei pesi $\boldsymbol{\theta}$ tramite il metodo della *Discesa Stocastica del Gradiente* (SGD); tale calcolo viene effettuato per K epoche prima di aggiornare effettivamente il vettore dei pesi e scartare il precedente. Un'epoca consiste in un *batch* di passi temporali per i quali è valutato l'aggiornamento; l'aggiornamento effettivamente applicato ai parametri è la media degli aggiornamenti valutati su un certo numero di *batches*. Tale procedimento si esegue al fine di non modificare i parametri sulla base di un orizzonte limitato di dati raccolti, garantendo maggiore generalità alla funzione di controllo finale [27].

2.2.3 Il Framework *Stable Baselines*

Gli algoritmi per la definizione dei problemi di apprendimento per rinforzo oggetto del lavoro sono sviluppati nel linguaggio di programmazione *Python* data la disponibilità di diverse librerie *open-source* che implementano algoritmi di *Machine Learning* allo stato dell'arte; in particolare viene utilizzato il framework *Stable Baselines 2* [31] derivato da *OpenAI gym* e costruito sulla libreria aperta *Tensorflow* versione 1.15 sviluppata da *Google Brain*.

Viene definita la classe di funzioni che implementano le equazioni del moto del modello del velivolo e vengono definiti gli strati di ingresso ed uscita delle reti neurali per ogni controllore da realizzare: tali strati costituiscono i punti di interfaccia tra il controllore e il modello di simulazione del quadricottero. Inoltre ogni classe contiene due funzioni principali: la funzione *reset()* che, richiamata all'inizio di ogni nuovo episodio, elabora lo stato iniziale del modello di quadricottero e la funzione *step()* che contiene le equazioni del moto del modello e implementa il metodo di integrazione.

Nel lavoro sono state scritte diverse classi per poter realizzare i tre controllori illustrati in precedenza e simulare diverse strategie di modellazione del quadricottero; in particolare sono stati implementati due modelli le cui caratteristiche vengono illustrate nel capitolo 3. Nel codice richiamato per avviare il processo di allenamento vengono inoltre definiti alcuni parametri dell'algoritmo denominati *hyperparameters*; il loro valore in ogni allenamento è stato scelto empiricamente partendo dai risultati del lavoro [5]. In particolare i parametri sono:

- N_{env}: numero di ambienti attivi in parallelo durante l'allenamento; è buona norma impostare un numero multiplo dei *thread* disponibili nel calcolatore che

esegue il calcolo. In questo modo si limita lo scambio di dati tra diversi *core* del processore e di conseguenza si riduce il tempo necessario all'esecuzione del calcolo;

- N_{steps}: numero di passi temporali per ogni *batch* per ogni agente parallelo; ad ogni passo temporale corrisponde una chiamata della funzione *step*;
- $N_{miniBatches}$: numero di aggiornamenti fatti in un *batch*, per modificare i pesi si effettua la media degli aggiornamenti calcolati in ogni *mini-batch*;
- $N_{optEpochs}$: l'aggiornamento è valutato per un numero di *batch* consecutivi pari al numero di epoche prima di essere applicato ai pesi e modificare dunque la *policy*;
- γ: fattore di sconto applicato nel calcolo del ritorno;
- α: il rateo di apprendimento è il parametro che moltiplica il vettore di modifica dei pesi; nel presente lavoro questo valore diminuisce linearmente all'avanzare dell'allenamento per limitare l'aggiornamento della *policy* e migliorare le prestazioni dell'algoritmo;
- ϵ: il valore del *clip-range* costituisce il limite massimo nella valutazione della funzione obiettivo per l'aggiornamento dei pesi; anche questo parametro presenta un andamento linearmente decrescente con il progresso dell'allenamento.

Capitolo 3

Modello dinamico del quadricottero

Il modello di simulazione è costruito sul velivolo *Hummingbird* prodotto dall'azienda tedesca *Ascending Technology GmbH* oggi confluita in Intel.

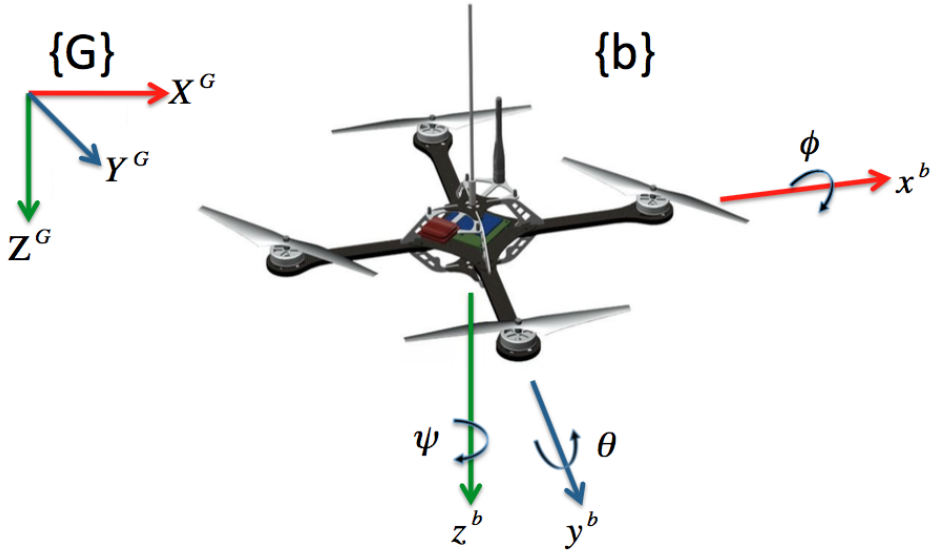

Figura 3.1. Ascending Technology Hummingbird e sistemi di riferimento; il sistema $X_G - Y_G - Z_G$ è il sistema di assi terra di seguito indicato con $X_E - Y_E - Z_E$.

E' stato scelto questo velivolo come base per la formulazione del modello in quanto presenta delle caratteristiche costruttive di semplice riproducibilità, con componenti economici e facilmente reperibili; tali caratteristiche consentono di sfruttare i risultati della presente trattazione in futuri lavori il cui obiettivo sarà la realizzazione di prove di volo di un quadricottero per il quale si sviluppa l'algoritmo di controllo con la tecnica del RL. I parametri costruttivi del quadricottero sono riportati in tabella 3.1, mentre in figura 3.2 è rappresentato uno schema della configurazione del mezzo visto dall'alto. Il mezzo è orientato in configurazione "+" con i motori disposti

nei quattro vertici di un quadrato, il rotore anteriore e quello posteriore (motori 1 e 3) presentano verso di rotazione antiorario, dunque la coppia reattiva esercitata dai rotori sul corpo del mezzo costituisce un contributo positivo al momento di imbardata, viceversa per i motori di destra e sinistra (2 e 4).
I rotori sono azionati da motori elettrici di tipo *bruchless*.

Parametro	Valore	note
Lx	0.34 m	distanza tra motori adiacenti
Ly	0.34 m	il drone ha base quadrata
massa	0.71 kg	MTOW assunto
massa batteria	0.186 kg	massa della batteria
massa corpo	0.484 kg	massa del corpo centrale
raggio sfera	0.06 m	raggio sfera corpo centrale
massa motore	0.04 kg	massa motore+elica+ESC+braccio

Tabella 3.1. Dati del modello di quadrotor

La parte contenente l'elettronica e il corpo centrale del *frame* è considerata, ai fini del calcolo dell'inerzia, come una sfera uniforme la cui massa e il cui raggio sono riportati in tabella 3.1. Per il calcolo dell'inerzia si considerano delle masse concentrate nel punto di posizionamento dei motori che rappresentano l'insieme di motore, elica, controllore di velocità e braccio del telaio. La batteria invece viene considerata posizionata sotto il corpo centrale, allineata con l'asse Z_B ad una distanza dal baricentro assunta pari a 0.06 m.

3.1 Ipotesi e finalità del modello di simulazione

Il modello di simulazione del quadricottero presenta due finalità; la prima è quella di caratterizzare l'ambiente da controllare nello stadio di definizione dei parametri delle funzioni di controllo: come anticipato nel capitolo introduttivo, i dati sui quali viene calcolato l'aggiornamento dei parametri della rete neurale sono raccolti impiegando la stessa sul sistema da controllare. In questo contesto la rete neurale elabora i comandi e attraverso il modello è calcolata l'evoluzione dello stato del velivolo, in seguito, sulla base dello stato cinematico elaborato, è calcolato il *guadagno* per l'aggiornamento dei parametri della rete neurale. Il secondo scopo del modello è quello di effettuare delle simulazioni con cui verificare le prestazioni dei controllori ottenuti.

L'approccio utilizzato per la modellazione è di tipo misto: si scrivono le equazioni del moto formulando le ipotesi necessarie alla descrizione delle azioni e, contemporaneamente, si fa riferimento ai risultati del lavoro di identificazione di Napoleoni in [17]. Il mezzo è considerato un corpo rigido, le variabili di controllo sono le velocità di rotazione dei quattro rotori. I motori elettrici usati dai quadricotteri di questa categoria sono di tipo *brushless* ovvero senza spazzole, controllati elettronicamente

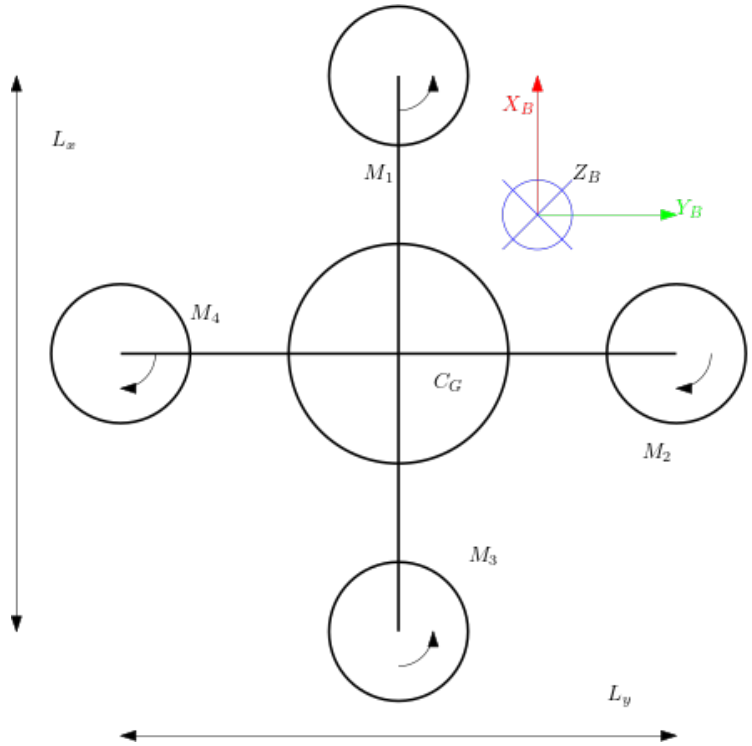

Figura 3.2. Vista 2D dall'alto (verso negativo di asse Z_b) del quadricottero di cui si è formulato il modello.

da dispositivi che ne regolano la velocità di rotazione in base al segnale ricevuto dal *flight controler*; non viene caratterizzata la dinamica del motore, la variazione di velocità angolare del rotore è considera istantanea rispetto al comando. Si assume che sia la spinta che la coppia assorbita dai rotori siano proporzionali al quadrato della velocità di rotazione. Per il mezzo in esame si considera un inviluppo di volo che prevede una velocità massima di 20 m/s e una quota massima di 50 m, pertanto si assume che il valore della densità sia costante e pari alla densità al livello del mare: $\rho_{SL} = 1.225$ kg/m^3.

Relativamente alle azioni aerodinamiche vengono trascurate le resistenze di forma e parassita esercitate dal corpo centrale e dai bracci del velivolo in quanto questi contributi assumono valori trascurabili nell'inviluppo di volo considerato. Non è invece possibile trascurare il comportamento aerodinamico dei rotori: le componenti di velocità verticali e nel piano producono effetti importanti sulla trazione generata dai rotori e inoltre comportano la comparsa di azioni nel piano. Sono formulati due modelli distinti per includere gli effetti appena citati: il primo modello è ricavato dalla trattazione nel testo [12] e prevede che la resistenza aerodinamica e la coppia resistiva siano proporzionali rispettivamente alle componenti di velocità e velocità angolare in assi corpo. Tale modello caratterizzato da un comportamento lineare è attendibile nella descrizione del velivolo solo in un intorno della condizione di volo a punto fisso.
In seguito, con riferimento alla trattazione proposta da Hoffmann et al. in [32]

si elabora il modello aerodinamico del rotore considerando l'effetto dell'angolo di influsso e del flappeggio. Lo scopo di questo secondo modello è quello di fornire una descrizione del quadricottero più accurata nell'inviluppo di volo considerato; per tale motivo questo modello è utilizzato nelle simulazioni di validazione dei controllori ottenuti. Il vettore della trazione è considerato sempre orientato secondo la direzione verticale. La motivazione che spinge alla formulazione di due modelli di diversa complessità è chiarita nel seguito.

Nel capitolo vengono esposte le equazioni del moto del velivolo e viene presentata la relazione tra le variabili di controllo e i comandi di spinta media e momenti in assi corpo elaborati dalla legge di controllo. E' riportata la formulazione della spinta e della coppia in relazione alla velocità angolare dei rotori e, in seguito, è presentata la modellazione aerodinamica del rotore con le distinzioni descritte in precedenza; infine è effettuata l'analisi del modello linearizzato e sono presentate le caratteristiche del metodo di integrazione del sistema di equazioni differenziali.

3.2 Sistemi di riferimento

I sistemi di riferimento utilizzati sono:

- Sistema corpo $[X_B, Y_B, Z_B]$: sistema di riferimento centrato nel baricentro del mezzo e solidale al corpo rigido, l'asse X_B è orientato verso il motore anteriore, l'asse Y_B verso il motore di destra e l'asse Z_B verso il basso;
- Sistema NED $[X_E, Y_E, Z_E]$: sistema di riferimento fisso rispetto ad un punto scelto come origine a terra, tale sistema è considerato solidale alla superficie terrestre e contemporaneamente inerziale, trascurando le accelerazioni angolari legate al moto di rotazione e rivoluzione terrestre; l'asse X_E è orientato verso Nord, l'asse Y_E verso Est e l'asse Z_E diretto secondo la verticale locale.

Le forze e i momenti vengono definiti in assi corpo in quanto in questa terna le componenti del tensore di inerzia sono costanti. La posizione del mezzo è riferita rispetto al sistema NED.

3.3 Equazioni del moto

Vengono riportate le equazioni della dinamica e della cinematica in forma vettoriale [33]

$$\begin{cases} m(\dot{\boldsymbol{V}}_B + \boldsymbol{\omega}_B \wedge \boldsymbol{V}_B) = \boldsymbol{F}_P + \boldsymbol{F}_A + \boldsymbol{F}_T + \boldsymbol{F}_D \\ \underline{I}\,\dot{\boldsymbol{\omega}}_B + \boldsymbol{\omega}_B \wedge \boldsymbol{I}\boldsymbol{\omega}_B = \boldsymbol{G}_T + \boldsymbol{G}_A + \boldsymbol{G}_D \\ \dot{\boldsymbol{Q}} = \frac{1}{2}\,\underline{\boldsymbol{\omega}_B}\,\boldsymbol{Q} \\ \dot{\boldsymbol{P}}_E = \boldsymbol{L}_{EB}\boldsymbol{V}_B \end{cases} \tag{3.1}$$

dove m è la massa del velivolo mentre il tensore di inerzia è una matrice diagonale in quanto si assume il quadricottero simmetrico rispetto ai piani principali

$$\boldsymbol{I} = \begin{bmatrix} I_{xx} & 0 & 0 \\ 0 & I_{yy} & 0 \\ 0 & 0 & I_{zz} \end{bmatrix}. \tag{3.2}$$

Il sistema consiste in 13 equazioni differenziali non lineari. Le variabili di stato del sistema sono le sei componenti di velocità e velocità angolare, la posizione e le quattro componenti del quaternione. I vettori di velocità e velocità angolare con coordinate in assi corpo sono:

$$\boldsymbol{V}_B = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \qquad\qquad \boldsymbol{\omega}_B = \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{3.3}$$

La posizione è descritta dalle coordinate del baricentro del quadricottero rispetto all'origine della terna NED con il vettore: $\boldsymbol{P}_E = [X_E,\ Y_E,\ Z_E]^T$. Il pedice E indica che le componenti sono espresse in assi terra. La rotazione rigida del sistema di riferimento corpo rispetto al sistema di riferimento NED è rappresentata tramite il quaternione, al fine di eliminare le singolarità legate all'uso degli angoli di Eulero. Un secondo vantaggio legato all'uso dei quaternioni è che la relazione tra le derivate temporali delle componenti del quaternione e le componenti della velocità angolare è lineare. Per il quaternione di rotazione si usa convenzionalmente la notazione:

$$\boldsymbol{Q} = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{3.4}$$

$$\begin{cases} q_0 = cos\frac{\mu}{2} \\ q_1 = sin\frac{\mu}{2}\ cos\alpha \\ q_2 = sin\frac{\mu}{2}\ cos\beta \\ q_3 = sin\frac{\mu}{2}\ cos\gamma \end{cases} \tag{3.5}$$

dove q_0 è la parte reale mentre q_1, q_2, q_3 sono le componenti della parte immaginaria ovvero le coordinate del versore dell'asse di rotazione della terna mobile, espresse nel riferimento fisso; μ è l'angolo di rotazione [34]. La matrice $4x4$ elaborata sulla base delle 3 componenti di velocità angolare e utile per calcolare le relazioni differenziali del quaternione ha la forma:

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix} = \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & \tilde{\boldsymbol{\omega}} \end{bmatrix} \tag{3.6}$$

Infine si riporta la relazione della matrice di trasformazione da assi corpo ad assi NED ($\boldsymbol{L}_{EB}$) sia nella rappresentazione con angoli di Eulero che nella rappresentazione tramite quaternione:

$$\boldsymbol{L}_{EB} = \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_0 q_2 + q_1 q_3) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_0 q_1 + q_2 q_3) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} = \tag{3.7}$$

$$\begin{bmatrix} \cos\theta\cos\psi & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\psi + \cos\phi\sin\theta\cos\psi \\ \cos\theta\sin\psi & \cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi & -\sin\phi\cos\psi + \cos\phi\sin\theta\sin\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \tag{3.8}$$

3.4 Modellazione di Forze e Momenti

I vettori di forza e momento sono espressi per componenti in assi corpo, tra le azioni agenti sul velivolo si considerano la forza peso F_P, la spinta generata dai quattro rotori F_T e la forza aerodinamica F_A; il momento esterno G_T invece è generato dalle azioni propulsive, mentre G_A rappresenta il contributo aerodinamico al momento. I termini F_D e G_D sono di disturbi che rappresentano turbolenze aerodinamiche di natura casuale, la cui evoluzione dinamica non è definita.
Il vettore della forza peso in assi corpo è ottenuto come segue:

$$\boldsymbol{F}_P = \boldsymbol{L}_{BE} \cdot m\boldsymbol{g}_E = m\boldsymbol{L}_{BE} \cdot \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} = m \cdot g \cdot \begin{bmatrix} 2(q_1 q_3 - q_0 q_2) \\ 2(q_0 q_1 - q_2 q_3) \\ q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix} \tag{3.9}$$

Infine si scrive il vettore della forza propulsiva ottenuto come somma della trazione generata dai quattro rotori nella direzione verticale Z_B (le T_i non sottolineate sono i moduli delle forze e non i vettori):

$$\boldsymbol{F}_T = -\begin{bmatrix} 0 \\ 0 \\ T_1 + T_2 + T_3 + T_4 \end{bmatrix}. \tag{3.10}$$

Il momento di controllo è dato dalla:

$$\boldsymbol{G}_T = \sum_{i=1}^{4} \boldsymbol{r}_i \wedge \boldsymbol{T}_i + \boldsymbol{C}_i \tag{3.11}$$

dove $\underline{T_i}$ è la trazione dell'i-esimo rotore mentre r_i è il vettore posizione del mozzo dell'i-esimo rotore rispetto al baricentro . sviluppando la 3.11 avendo avendo assunto che i rotori siano sullo stesso piano del baricentro e che la trazione generate sia allineata con l'asse Z_B si ottiene:

$$\boldsymbol{G}_T = \begin{bmatrix} \frac{L_y}{2}(T_4 - T_2) \\ \frac{L_x}{2}(T_1 - T_3) \\ (C_1 + C_3 - C_2 - C_4) \end{bmatrix} \tag{3.12}$$

C_i rappresenta la coppia trasmessa dall'i-esimo rotore all'albero e scaricata nel punto di fissaggio tra il motore e il telaio del mezzo. La coppia è legata alla resistenza aerodinamica della pala e ha verso opposto al verso di rotazione dell'elica. Secondo

quanto riportato nella 3.12 si assume che, viste dall'alto verso il basso, le eliche 1 e 3 ruotano in senso antiorario mentre 2 e 4 ruotano in senso orario. E' trascurato il contributo giroscopico delle parti rotanti in quanto queste hanno massa piccola rispetto a quella dell'intero velivolo.

Comandi

Le variabili di controllo del mezzo (*pseudo*-comandi) che sono:

- δT: comanda un aumento o una diminuzione di pari valore della velocità di rotazione di ogni rotore, controlla la trazione complessiva;
- δE: comando sul momento di beccheggio, controlla la differenza di trazione tra il rotore anteriore e quello posteriore per generare un momento di beccheggio;
- δA: comando sul momento di rollio, controlla la differenza di trazione tra rotore di destra e sinistra per generare un momento di rollio;
- δR: comando sul momento di imbardata, controlla la differenza di trazione tra rotori che girano in senso orario (2, 4) e rotori che girano in senso antiorario (1, 3), ne consegue un momento di imbardata.

Le azioni di controllo vengono in seguito tradotte in un valore del comando per ogni motore.
Si calcola in primo luogo il valore del comando sulla trazione media:

$$\delta T_{av} = \delta T_{trim} \cdot (1 + K_1 \delta T) \tag{3.13}$$

Il comando δT controlla una variazione della trazione del singolo rotore rispetto al valore calcolato per la condizione di volo a punto fisso; questo valore impone la trazione T di un rotore pari a:

$$T = \frac{P}{4} \tag{3.14}$$

con $P = 6.97\,N$ peso del mezzo. Il numero di giri nella condizione di volo a punto fisso è calcolato in base alle prestazioni dei rotori. Il comando sulla velocità di rotazione di ogni rotore viene calcolato come [12]:

$$\begin{bmatrix} \delta T_{M1} \\ \delta T_{M2} \\ \delta T_{M3} \\ \delta T_{M4} \end{bmatrix} = \begin{bmatrix} \delta T_{av} + SF \cdot (\delta E + \delta R) \\ \delta T_{av} + SF \cdot (-\delta A - \delta R) \\ \delta T_{av} + SF \cdot (-\delta E + \delta R) \\ \delta T_{av} + SF \cdot (\delta A - \delta R) \end{bmatrix} \tag{3.15}$$

I parametri adimensionali valgono: $SF = 0.35$ e $K_1 = 0.6$; il valore di questi parametri è stato scelto su base empirica, facendo riferimento alle prestazioni ottenute dalle leggi di controllo nelle simulazioni di validazione. Alla funzione segue un controllo che limita il comando per ogni motore nell'intervallo $[0, 1]$.
Occorre precisare che un comando sul momento di rollio o di beccheggio non influenza il momento di imbardata agendo solo sulla differenza di spinta; questa caratteristica consente di disaccoppiare le dinamiche di assetto da quella direzionale nella fase

di linearizzazione e analisi modale a livello di azioni di controllo. Si sottolinea che nel presente lavoro si ipotizza che le azioni aerodinamiche sono disaccoppiate tra dinamiche di assetto e dinamica direzionale.

3.5 Modello del rotore

I motori del quadricottero sono di tipo *brushless* e sono guidati da controllori elettronici che ricevono il segnale dal *flight controller* relativo alla velocità angolare desiderata; questa tipologia di motori presenta un circuito di induzione elettromagnetica trifase sullo statore e dei magneti permanenti sul rotore. Il loro principio di funzionamento è analogo a quello dei motori elettrici asincroni trifase: il controllore elettronico genera i segnali elettrici ai capi degli avvolgimenti di statore necessari a creare il campo magnetico rotante che muove il rotore. La velocità di rotazione del campo magnetico di statore è calcolata dal controllore elettronico per mantenere il riferimento sulla velocità di rotazione tramite un sistema di controllo in retroazione. I rotori sono bipala con diametro per passo pari a 8x3.8 in, il numero di giri massimo ammesso per i rotori è $\Omega_{maxPROP} = 8550$ RPM $= 895.35$ rad/s mentre i coefficienti di prestazioni dell'elica sono pari a: $K_f = 5.70 \cdot 10^{-6}$ N s^2 e $K_q = 8.15 \cdot 10^{-8}$ N m s^2 nella condizione di volo a punto fisso, i valori sono ricavati dai risultati del lavoro di identificazione del modello del quadricottero in esame riportato in [17]. Le prestazioni dei rotori dipendono dalla condizione di volo: una velocità di traslazione del rotore comporta l'insorgenza di effetti aerodinamici che influenzano sia il modulo che la giacitura del vettore di trazione. Nel presente lavoro si ipotizza che i coefficienti di proporzionalità riportati siano costanti in tutto l'inviluppo di volo; gli effetti della velocità di traslazione del rotore vengono inclusi separatamente sotto forma di azioni aerodinamiche le cui caratteristiche sono discusse nel seguito. Nel modello la trazione dipende dal quadrato della velocità di rotazione dei rotori secondo la relazione:

$$\begin{aligned} F &= K_f \cdot \Omega_{prop}^2 \\ Q &= K_q \cdot \Omega_{prop}^2 \end{aligned} \tag{3.16}$$

da cui si ricava la relazione tra trazione e coppia assorbita:

$$K_{qf} = \frac{Q}{F} = \frac{K_q}{K_f} \tag{3.17}$$

con $K_{qf} = 1.42 \cdot 10^{-2}$ m.
La 3.16 permette di calcolare le azioni dei rotori a partire dal comando elaborato dalla legge di controllo.
Come anticipato, i motori del velivolo presentano un controllore elettronico che regola la velocità di rotazione, si assume che la variazione di velocità sia istantanea e che l'errore a stazionario rispetto al riferimento sia nullo. La velocità angolare dell'i-esimo rotore è calcolata come:

$$\Omega_{prop,i}^2 = \Omega_{maxPROP}^2 \cdot \delta T_{Mi} \tag{3.18}$$

dove si fa riferimento ai comandi sul numero di giri elaborati nella 3.15. Secondo la 3.18 il valore della trazione e della coppia calcolati con la 3.16 presentano andamento lineare rispetto al comando elaborato dalla legge di controllo; questa caratteristica è scelta per accelerare la definizione dei parametri delle funzione di controllo attraverso l'applicazione dell'algoritmo di RL. La relazione lineare tra comando e azione sul modello del velivolo consente di identificare con maggiore efficacia le funzioni di trasferimento del modello e accelera il processo che conduce alla definizione della legge di controllo che massimizza il guadagno spiegato nel capitolo precedente.

3.5.1 Modello aerodinamico del rotore

In letteratura raramente vengono considerati gli effetti della velocità di traslazione modellazione dei rotori dei quadricotteri per il design di sistemi di controllo e aumento di stabilità per questi mezzi, ciò perché si considerano condizioni prossime al volo a punto fisso; di conseguenze non è disponibile una formulazione standard per la caratterizzazione di tali effetti. Nel presente lavoro si fa riferimento ai risultati della modellazione dei rotori degli elicotteri riportando in particolare due fenomeni [32]: la variazione della trazione causata dalla variazione dell'angolo d'influsso e l'inclinazione del disco del rotore dovuta al flappeggio delle pale nel volo in traslazione. La varizione dell'angolo di influsso provoca variazioni dell'angolo di attacco della pala da cui dipende il valore assoluto della trazione, invece il flappeggio provoca l'inclinazione del vettore della trazione rispetto all'asse Z_B cui consegue la generazione di forze nel piano orizzontale.

In primo luogo viene calcolata la velocità relativa del rotore i-esimo in un riferimento centrato nel mozzo del rotore e orientato parallelamente agli assi corpo del velivolo; essendo il rotore solidale al corpo, la velocità $\boldsymbol{V}_{m,i}$ è valutata come:

$$\boldsymbol{V}_{m,i} = \boldsymbol{V}_b + \boldsymbol{\omega}_b \wedge \boldsymbol{R}_{m,i} \tag{3.19}$$

dove compaiono velocità e velocità angolare in assi corpo e il vettore posizione del motore rispetto al baricentro. Dalla formula precedente si evince come si generano velocità di traslazione lineari sul rotore anche in seguito a rotazioni rigide del quadricottero.

Le azioni aerodinamiche elaborate includendo gli effetti citati contribuiscono ad incrementare l'accuratezza del modello di simulazione ma introducono delle relazioni non lineari tra le forzanti e le variabili cinematiche. Nell'applicazione del RL per la definizione dei parametri delle leggi di controllo è opportuno che il modello su cui è applicata la rete neurale presenti una formulazione il più semplice possibile. Pertanto è necessario raggiungere un compromesso tra accuratezza del modello e complessità dello stesso per garantire che la funzione di controllo sia efficace e al tempo stesso ridurre il tempo necessario alla definizione dei parametri. Pertanto è formulato un modello caratterizzato da azioni aerodinamiche che presentano andamento lineare rispetto alle componenti di velocità angolare, tale formulazione fa riferimento ai lavori [12, 17, 35]; le caratteristiche di questo modello sono riportate a valle della trattazione su flappeggio ed angolo di influsso.
Si chiarisce che il modello per i motivi esposti, il modello che presenta andamento

lineare delle azioni aerodinamiche rispetto alle variabili cinematiche è applicato nello stadio di definizione dei parametri delle leggi di controllo, mentre il modello che include l'effetto del flappeggio e della variazione dell'angolo di influsso è usato per le simulazioni di validazione delle prestazioni delle leggi di controllo ottenute.

3.5.2 Modello del flappeggio

Il flappeggio è il grado di liberà rotazionale della pala di un rotore attorno ad un asse ortogonale sia all'asse della pala che all'asse dell'albero motore. Le pale dei rotori del quadricottero a cui si fa riferimento sono flessibili e formano un corpo continuo con il mozzo, pertanto le azioni che governano l'evoluzione dinamica del moto di flappeggio sono la portanza generata dalla pala, la forza di richiamo elastico del materiale nella sezione di incastro della pala e la forza centrifuga. Il moto di flappeggio determina la rotazione del *Tip Path Plane*(TPP) rispetto al piano ortogonale all'albero. Se si assume che la trazione generata da un rotore sia ortogonale al TPP, una sua rotazione modifica la giacitura della traizione. Per la descrizione del flappeggio si fa riferimento alla trattazione in [11].
Nel presente modello l'effetto del flappeggio viene incluso senza considerare il transitorio, l'effetto di una variazione della velocità di traslazione sulla rotazione del TPP è assunto istantaneo; tale ipotesi è supportata dal fatto che l'evoluzione del moto di flappeggio avviene con frequenze proprie pari alla velocità angolare del rotore, pertanto il tempo caratteristico della dinamica è motlo minore rispetto al tempo caratteristico più breve della dinamica del mezzo. Contrariamente agli elicotteri i rotori usati nei quadricotteri sono di piccole dimensioni rispetto al corpo del velivolo e presentano una struttura rigida, come anticipato il grado di libertà di flappeggio non è consentito dalla presenza di una cerniera, ma bensì dall'elasticità del materiale; la presenza di un angolo di flappeggio comporta dunque la generazione di un momento elastico di richiamo nella sezione di incastro della pala nel mozzo. A causa di queste caratteristiche costruttive l'angolo di rotazione del TPP ha entità piuttosto limitata; tuttavia nei lavori [32, 36] i ricercatori sottolineano come l'inclusione di tale effetto nella modellazione migliori le prestazioni di controllori sviluppati con tecniche *model-based* soprattutto in condizioni di velocità di traslazione elevata.

Dinamica di flappeggio

Lo schema della figura 3.3 è visualizzato il grado di libertà di flappeggio per una pala collegata al mozzo da una cerniera distante dall'asse dell'albero eR, dove e è un parametro adimensionale e R è il raggio del rotore; Ω è la velocità di rotazione della pala. L'evoluzione dinamica dell'angolo di flappeggio è descritta dalla relazione:

$$\ddot{\beta} + \Omega^2(1+\epsilon)\beta = M_A/B \quad con \quad \epsilon = M_b e x_g R^2/B \tag{3.20}$$

valida per angoli piccoli. M_A è il momento aerodinamico agente sulla pala, B è il momento di inerzia della pala rispetto alla cerniera, M_b è la massa della pala e x_g è la distanza non dimensionale del baricentro della pala dalla cerniera. Il secondo termine al primo membro della 3.23 rappresenta una rigidezza legata alla forza centrifuga esercitata sul baricentro della pala. Il momento aerodinamico vale:

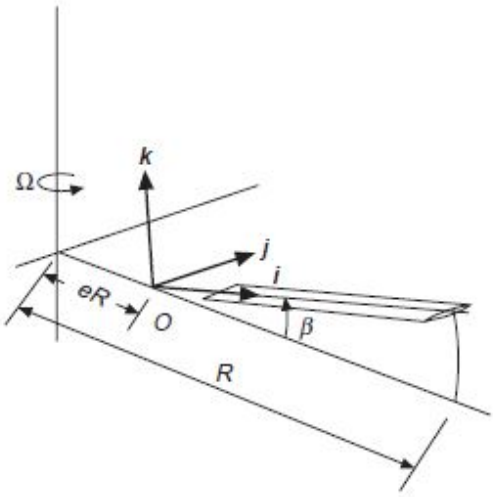

Figura 3.3. Grado di libertà di flappeggio di una pala [11].

$$M_A = \int_{blade} r \cdot dL = \int_0^{R_prop} \frac{1}{2}\rho c \Omega^2 r^3 C_L dr \tag{3.21}$$

dove c è la corda della pala, C_L è il coefficiente di portanza, $1/2\rho\Omega^2 r^2$ è la pressione dinamica ad una distanza r dal centro del rotore.

Nel caso di rotore flessibile, privo di cerniere di flappeggio, come per il quadricottero, va inclusa nell'equazione del moto 3.23 la forza di richiamo elastico esercitata dal materiale nella sezione di incastro della pala. Si assume che l'azione di richiamo elastico sia equivalente a quella di una molla torsionale di costante k_b; la coppia di richiamo si calcola come:

$$Q_{el} = k_b \beta \tag{3.22}$$

dove $k_b = 0.23$ Nm/rad [32] per il rotore considerato nel lavoro. Inserendo tale termine nella 3.23 si ottiene:

$$\ddot{\beta} + (\Omega^2(1+\epsilon) + \frac{k_b}{B})\beta = M_A/B \tag{3.23}$$

Nel momento aerodinamico M_A è incluso il contributo legato alla velocità di flappeggio $\dot{\beta}$ che comporta una variazione dell'angolo di attacco della pala che produce un'azione di smorzamento. La velocità di traslazione costituisce un elemento forzante ciclico per la pala con frequenza pari a Ω. La pala avanzante sperimenta infatti una velocità pari a $\Omega r + V_{rel}$ mentre la pala retrocedente una velocità pari a $/Omegar - V_{rel}$; ciò comporta una variazione della pressione dinamica tra pala avanzante e retrocedente cui consegue la generazione di una componente armonica del momento aerodinamico con frequenza pari alla velocità angolare.
Per semplificare la soluzione dell'equazione 3.23, si passa dal dominio del tempo a quello della posizione angolare della pala sul giro Ψ con:

$$d\Psi = \Omega dt \tag{3.24}$$

In seguito si scrive l'angolo di flappeggio sul giro con lo sviluppo in serie di Fourier, trascurando i termini di ordine superiore al primo:

$$\beta(\Psi) = a_0 - a_1 cos\Psi - b_1 sin\Psi \tag{3.25}$$

La posizione $\Psi = 0$ corrisponde al retro del rotore. Sostituendo la 3.25 nella 3.23 si ottiene un problema algebrico nelle incognite a_0, a_1 e b_1 dette coefficienti di flappeggio.

Inclusione nel modello

Nel modello esposto in questo lavoro si trascura il transitorio della dinamica di flappeggio, considerando solo la risposta a stazionario. A causa dello sfasamento tra forzante e risposta, la velocità di traslazione lungo l'asse X_M (u_M) del motore provoca una rotazione del TPP di un angolo a_1 attorno all'asse Y_M; la velocità lungo Y_M (v_M) provoca una rotazione del TPP pari a b_1 attorno a X_M. La figura 3.4 riporta quanto spiegato.

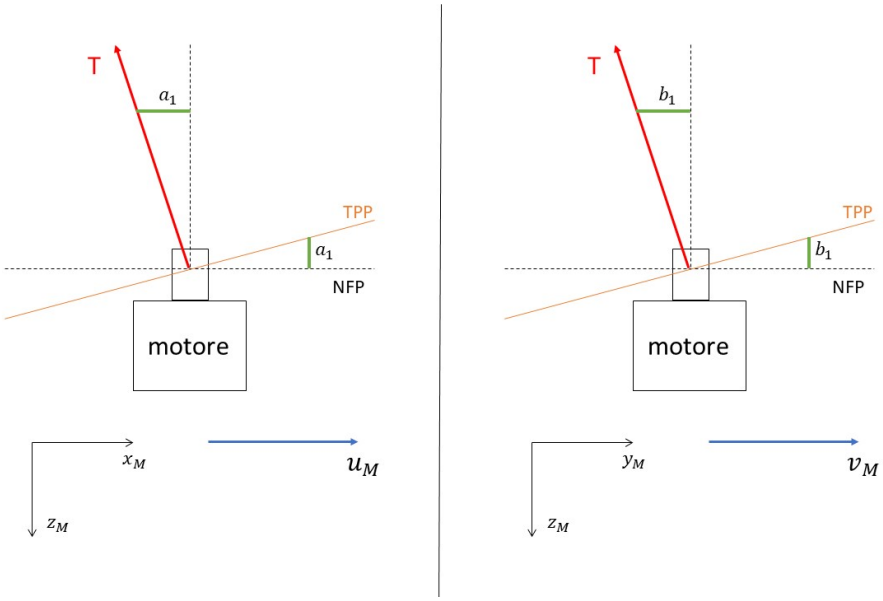

Figura 3.4. Effetto del flappeggio legato a velocità positive del motore lungo gli assi x_M e y_M.

I coefficienti di flappeggio a_1 e b_1 sono calcolati come [11, 32]:

$$a_1 = \frac{2\mu_x\left(\frac{4}{3}\theta_0 + \lambda_i\right)}{(1 - \mu_x^2/2)} \quad e \quad b_1 = \frac{2\mu_y\left(\frac{4}{3}\theta_0 + \lambda_i\right)}{\left(1 - \mu_y^2/2\right)} \tag{3.26}$$

I termini μ_x e μ_y sono il rapporto rispettivamente tra u_M e v_M rispetto alla velocità al tip $\Omega D_{prop}/2$, θ_0 è il passo della pala che in questa trattazione viene

assunto pari al passo riportato nel *datasheet* dell'elica mentre λ_i è il termine di velocità verticale relativa al tip, scontata della velocità indotta. Le formule 3.26 permettono di calcolare i coefficienti di flappeggio a stazionario in presenza di velocità di traslazione; non è considerato il contributo dell'angolo di conicità perchè i suoi effetti si annullano tra i quattro rotori dato che ruotano a coppie in versi opposti. Va sottolineato che le relazioni proposte permettono di valutare l'effetto del flappeggio sul TPP nel caso di rotore articolato; il rotore del quadricottero considerato è flessibile e non presenta una cerniera di flappeggio. Trascurando il contributo della coppia di richiamo elastico si elabora una sovrastima della rotazione del TPP: un confronto tra l'andamento calcolato con le 3.26 e la rotazione effettivamente misurata in [32] per un rotore come quello considerato è riportato nel grafico in figura 3.5.

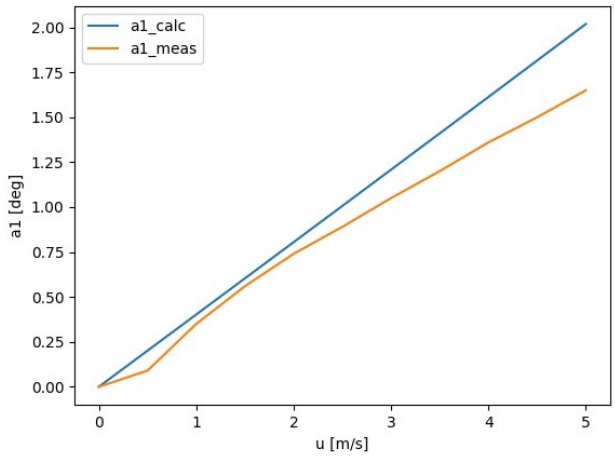

Figura 3.5. Confronto tra il coefficiente di flappeggio a_1 in gradi al variare della velocità in direzione X_M calcolato con la 3.26 e la rotazione misurata in [32].

Una traslazione in direzione positiva di X_B comporta una rotazione positiva secondo lo schema in figura 3.4; consegue la generazione di una forza che si oppone al moto. L'andamento del coefficiente b_1 è identico a quello di a_1 data la simmetria del mezzo.
Lo spostamento del vettore di trazione rispetto alla direzione di Z_B comporta la generazione di azioni nel piano che vengono aggiunte al computo delle forze esterne con la formula: $T_{in\,plane} = -sin(a_1) \cdot T$. La trazione in direzione verticale si assume sempre pari a T considerando dunque $cos(a_1) = cos(b_1) \cong 1$.

3.5.3 Modello dell'influsso

L'angolo di influsso corrisponde alla variazione di angolo di attacco della pala di un rotore causato dalla velocità indotta e dalla velocità di rotazione; l'angolo di attacco vale infatti:

$$\alpha \cong \theta - \frac{V_c + v_i}{\Omega\, r} \tag{3.27}$$

dove θ è il passo della pala, V_c è la velocità di traslazione del rotore che dà un contributo negativo se il rotore trasla verso l'alto o positivo se la traslazione è verso l'alto, v_i è la velocità indotta dal rotore e Ω è la velocità di rotazione della pala;

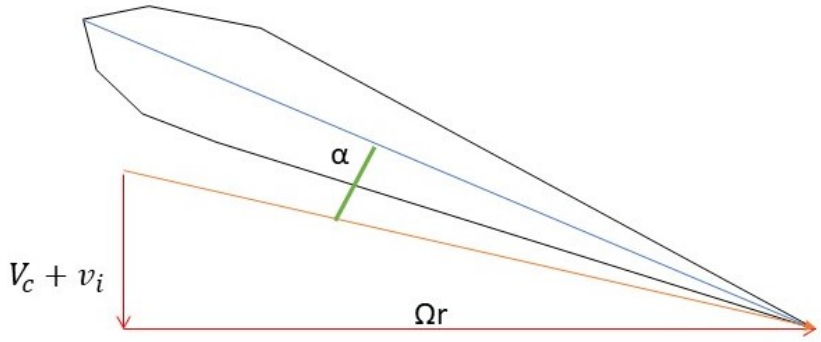

Figura 3.6. Effetto dell'influsso sull' angolo d'attacco della pala.

Teoria del disco attuatore

Per il calcolo della velocità indotta si fa rifermento alla teoria del disco attuatore seguendo la trattazione in [11]; questo modello richiede un'ipotesi fondamentale ovvero che la variazione di pressione e velocità è uniformemente distribuita sul disco del rotore. Questa ipotesi è piuttosto restrittiva ma viene accettata per rotori di piccole dimensioni come quelli utilizzati dal quadricottero oggetto del lavoro.
Si fa riferimento ad un tubo di flusso che comprende il disco del rotore di area A_d, una zona indisturbata a monte caratterizzata dalla velocità di salita V_c e dalla pressione p_{inf} e una zona a valle caratterizzata da velocità $V_c + v_2$ e pressione pari a p_{inf}. Il rotore genera una spinta positiva verso l'alto e impone una variazione di pressione pari a Δp mentre la velocità nella zona del disco vale $V_c + v_i$; la densità ρ è assunta costante.

La spinta è pari a:

$$T = \dot{m}(V_c + v_2) - \dot{m}V_c \quad con \quad \dot{m} = \rho A_d(V_c + v_i) \tag{3.28}$$

e contemporaneamente:

$$T = A_D \cdot \Delta p \tag{3.29}$$

con A_D area del disco del rotore e Δp salto di pressione tra monte e valle del rotore. Si considera la portata in massa $\dot{m}$ costante nel tubo di flusso. Sostituendo e risolvendo per v_2 dopo aver calcolato il salto di pressione con l'equazione di Bernoulli si ottiene:

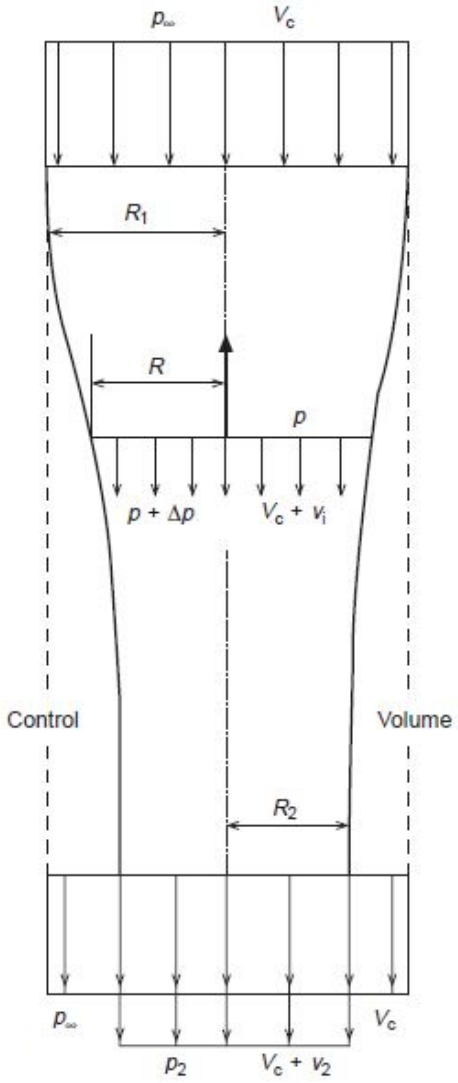

Figura 3.7. Schema del tubo di flusso per la Teoria del disco attuatore da [11].

$$v_2 = 2v_i \tag{3.30}$$

La spinta infine è pari a:

$$T = 2A_d \rho v_i (V_c + v_i) \tag{3.31}$$

Si valuta la velocità di trazione v_h corrispondente alla velocità indotta per $V_c = 0$ per il mezzo oggetto del lavoro: la spinta in condizioni di volo a punto fisso deve essere pari ad un quarto del peso.

$$\begin{aligned} v_h &= \sqrt{\frac{T}{2 \cdot \rho A_d}} = 4.68\, m/s \\ T &= \frac{P}{4} = 1.742\, N \end{aligned} \tag{3.32}$$

Si scrivono ora la velocità di salita ed indotta non dimensionale:

$$\overline{V_c} = \frac{V_c}{v_h} \quad e \quad \overline{v_i} = \frac{v_i}{v_h} \tag{3.33}$$

Sostituendo nella 3.31 si ottiene:

$$(\overline{V_c} + \overline{v_i})v_i - 1 = 0 \tag{3.34}$$

da cui:

$$v_i = -\frac{V_c}{2} + \sqrt{\left(\frac{V_c}{2}\right)^2 + v_h^2} \tag{3.35}$$

E' necessario sottolineare che il modello appena proposto è accurato nel valutare la velocità indotta solo in condizioni di volo normali per cui $V_c/v_h \geq -0.4$; pertanto sono coperte anche condizioni in cui la velocità è opposta al verso della trazione. Per velocità minori rispetto al limite citato si incorre nella condizione di *vortex ring state* (VRS); tale condizione comporta la generazione di vortici in corrispondenza del rotore per i quali diventa complesso elaborare un modello [32]. Nel presente lavoro non si ipotizza che il quadricottero non raggiunga mai la condizione di VRS.

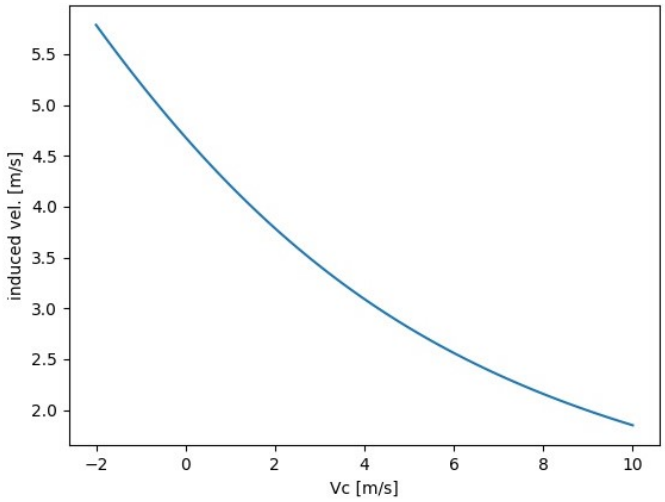

Figura 3.8. Andamento della velocità indotta rispetto alla velocità di salita.

Nel grafico 3.8 è riportato l'andamento della velocità indotta all'aumentare della velocità di salita; Si assume velocità di salita positiva se concorde in verso con la trazione generata e velocità indotta positiva se discorde in verso con la trazione generata. Il grafico è realizzato per un valore costante della manetta.

Contributo della velocità indotta sulla spinta

Il contributo della velocità indotta è inserito nel modello come una variazione della spinta a punto fisso attraverso la teoria dell'elemento di pala:

$$T_{tot} = T_H + \Delta T \tag{3.36}$$

I dati prestazionali dell'elica per la spinta e la coppia a punto fisso includono l'effetto della velocità di trazione. La variazione di spinta legata alla variazione dell'angolo di attacco rispetto al volo a punto fisso è calcolata come:

$$\Delta T = \frac{1}{4} abc\Omega R^2 (V_c + v_i - v_h) \tag{3.37}$$

dove a è la derivata del coefficiente di portanza della pala rispetto all'angolo d'attacco $a = dC_L/d\alpha$ assunta costante, b è il numero di pale del rotore pari a 2, c è la corda media della pala e Ω la velocità angolare del rotore. L'ipotesi che supporta la valutazione della spinta come nella 3.36 è che il coefficiente di portanza della pala è lineare al variare dell'angolo di attacco.

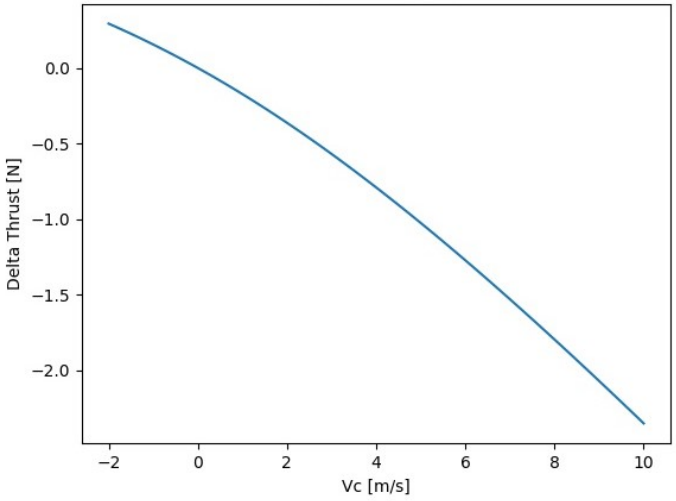

Figura 3.9. Andamento del contributo ΔT all'aumentare della velocità di salita.

In figura 3.9 è riportato l'andamento della variazione di spinta calcolata con la 3.37 al variare della velocità di salita; un valore negativo di questo contributo si traduce in una diminuzione della spinta complessiva fornita dal motore. Il grafico è realizzato lasciando invariato il valore della manetta.
Come evidente nel grafico 3.9 il contributo valutato con il metodo presentato introduce un aumento della trazione complessiva nel caso in cui la velocità di traslazione del rotore ha verso opposto a quello della trazione e viceversa; tale effetto è legato alla variazione dell'angolo di attacco della pala. Nel presente lavoro si considera che, oltre alla velocità di traslazione w del velivolo, anche le componenti di velocità angolare in assi corpo p e q producono effetti sull'influsso: ad esempio in presenza di una velocità di rollio positiva, il rotore di destra vede una velocità di traslazione verso il basso mentre quello di sinistra verso l'alto; ciò comporta un aumento della trazione del rotore di destra e una diminuzione della trazione del rotore di sinistra, cui consegue un'azione smorzante rispetto alla velocità di rollio.

3.5.4 Modello con contributi lineari di resistenza aerodinamica

Come anticipato è formulato un secondo modello aerodinamico del quadricottero che prevede che i contributi aerodinamici sulle forze e i momenti hanno andamento lineare

rispetto alla velocità e velocità angolare in assi corpo; queste azioni si riferiscono agli effetti della velocità di traslazione del rotore sulla trazione in direzione Z_B e sulla generazione di componenti di componenti di forza *in-plane*. Diversi lavori sfruttano questa tipologia di modellazione delle azioni aerodinamiche dei rotori al fine di ottenere una caratterizzazione aerodinamica semplice ma che descriva con un certo grado di accuratezza la dinamica del velivolo in un intorno della condizione di volo a punto fisso [12, 37, 35, 38]. La resistenza aerodinamica alla rotazione e alla traslazione valgono:

$$\boldsymbol{F}_A = -\begin{bmatrix} C_{dX}u \\ C_{dY}v \\ C_{dZ}w \end{bmatrix} \tag{3.38}$$

con C_{dX}, C_{dY} e C_{dZ} [kg/s] sono i coefficienti di resistenza riferiti alle componenti di velocità lineare. La coppia resistiva dovuta alla velocità angolare è calcolata come:

$$\boldsymbol{G}_A = -\begin{bmatrix} C_{dRX}p \\ C_{dRY}q \\ C_{dRZ}r \end{bmatrix} \tag{3.39}$$

in cui C_{dRX}, C_{dRY} e C_{dRZ} sono i coefficienti di resistenza riferiti alle componenti di velocità angolare. Il valore dei coefficienti sopra esposti è stato calcolato a partire dai contributi del flappeggio e dell'influsso valutati in un intorno della condizione di volo in *hovering*. I valori specifici adottati sono :

$$\begin{bmatrix} C_{dX} \\ C_{dY} \\ C_{dZ} \end{bmatrix} = \begin{bmatrix} 0.05 \\ 0.05 \\ 0.1 \end{bmatrix} \; kg/s \tag{3.40}$$

$$\begin{bmatrix} C_{dRX} \\ C_{dRY} \\ C_{dRZ} \end{bmatrix} = \begin{bmatrix} 0.02 \\ 0.02 \\ 0.001 \end{bmatrix} \; kg\,m^2 \tag{3.41}$$

Tali valori sono stati ottenuti partendo da alcuni risultati del lavoro di identificazione di Napoleoni in [17] e dalla caratterizzazione degli effetti sopra citati nella condizione di volo a punto fisso.
Il modello con aerodinamica lineare è accettabile nell'intorno della condizione di equilibrio di volo a punto fisso; per tale motivo non viene usato nelle simulazioni di validazione delle prestazioni dei controllori ottenuti. Tuttavia, come anticipato, risulta conveniente usare questo modello come ambiente da controllare su cui eseguire la procedura di definizione dei parametri delle funzioni di controllo. Infatti l'aggiornamento dei parametri della *policy* avviene con riferimento alle funzioni di trasferimento dell'ambiente e, al fine di abbreviare il tempo necessario per completare l'algoritmo, è opportuno rendere tali relazioni il più semplici possibile, compiendo una scelta di compromesso tra tempo a disposizione per l'algoritmo e accuratezza del modello dell'ambiente.

3.6 Linearizzazione del modello dinamico

A partire dal modello con resistenza aerodinamica lineare rispetto alla velocità e velocità angolare si procede alla linearizzazione al fine di eseguire l'analisi modale del sistema. Data la semplicità degli effetti aerodinamici considerati non emergono modi oscillatori né dinamiche instabili; tale analisi è di interesse per valutare i tempi caratteristici del sistema in esame e scegliere di conseguenza il passo di integrazione appropriato. L'analisi è svolta riportando il modello in ambiente Simulink ed effettuando la linearizzazione con Control System Designer; si riportano i risultati disaccoppiando le dinamiche riferite alle variabili di assetto longitudinale e laterale e di traslazione verticale dalla dinamica riferita alla velocità di imbardata.

3.6.1 Linearizzazione di assetto e traslazione verticale

Il vettore di stato considerato è $\underline{x} = [u, v, w, p, q, \phi, \theta]^T$ mentre i controlli perturbati sono $\underline{u} = [\delta T, \delta E, \delta A]$.
La matrice di stato A vale:

$$A_{Att} = \begin{bmatrix} -0.0704 & 0 & 0 & 0 & 0 & 0 & -9.815 \\ 0 & -0.0704 & 0 & 0 & 0 & 9.815 & 0 \\ 0 & 0 & -0.7042 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -5.4370 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -5.4370 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{3.42}$$

con autovalori: $\lambda_1 = \lambda_2 = -0.0704$; $\lambda_3 = 0.704$; $\lambda_4 = \lambda_6 = 0$; $\lambda_5 = \lambda_7 = -5.4370$, come atteso gli autovalori hanno tutti parte reale minore o uguale a 0 e parte immaginaria nulla rappresentando dunque moti non oscillatori stabili asintoticamente; gli autovettori corrispondenti agli autovalori sopra elencati sono le colonne della seguente matrice:

$$\underline{\underline{Z}} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & -1 & -0.3141 \\ 0 & 1.0000 & 0 & 1.0000 & 0.3141 & 0 & 0 \\ 0 & 0 & 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.9337 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.9337 \\ 0 & 0 & 0 & 0.0072 & -0.1717 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0072 & -0.1717 \end{bmatrix} \tag{3.43}$$

I tempi di dimezzamento per le dinamiche sopra riportate sono: $T_{halv,\lambda_{1,2}} = 9.8452\ s$, $T_{halv,\lambda_3} = 0.9842\ s$, $T_{halv,\lambda_{5,7}} = 0.1275\ s$. Dalla matrice degli autovettori si deduce che gli autovalori cui corrisponde il tempo di dimezzamento più breve sono riferiti alla risposta ad una perturbazione delle velocità di rollio e beccheggio.

3.6.2 Linearizzazione della dinamica di rotazione all'imbardata

Si effettua la linearizzazione della dinamica riferita al grado di libertà di rotazione all'imbardata; lo stato considerato è: $\underline{x} = [r, \psi]^T$ e perturbando il controllo $\underline{u} = \delta R$. La matrice di stato in questo caso vale:

$$A_{Dir} = \begin{bmatrix} -0.1006 & 0 \\ 1 & 0 \end{bmatrix} \tag{3.44}$$

a cui corrispondono gli autovalori: $\lambda_1 = 0$; $\lambda_2 = -0.1006$; che indicando modi non oscillatori stabili asintoticamente. Si riporta la matrice degli autovettori:

$$\underline{\underline{Z}} = \begin{bmatrix} 0 & 0.1001 \\ 1.0000 & -0.9950 \end{bmatrix} \tag{3.45}$$

Il tempo di dimezzamento riferito all'autovalore non nullo vale: $T_{halv,\lambda_2} = 6.8897\ s$. Dalla matrice degli autovettori si deduce che questo parametro descrive la risposta ad una perturbazione della velocità di imbardata e dell'angolo di rotta

3.7 Integrazione

Le equazioni differenziali 3.1 costituiscono un sistema di ODE del tipo $\dot{\underline{x}} = \underline{f}(\underline{x}, \underline{u})$ integrato con il metodo a passo fisso h di Runge-Kutta al quarto ordine. Alla luce di quanto emerso dall'analisi linarizzata, il tempo caratteristico più breve è quello relativo alle dinamiche di rollio e beccheggio che vale: $T_{halv,\lambda_{5,7}} = 0.1275\ s$; pertanto si sceglie un passo di integrazione $h = 0.01s$ tenendo conto del vincolo di natura empirica di impostare un tempo di integrazione almeno 10 volte più piccolo del tempo caratteristico più breve.
L'allenamento della *policy* viene eseguito raccogliendo dati attraverso l'applicazione diretta della stessa sul modello per un determinato numero di passi temporali; al fine di ottimizzare l'uso delle risorse computazionali e ridurre il tempo di allenamento, è necessario impostare l'intervallo tra due valutazioni consecutive delle azioni da parte della *policy* più lungo possibile. La motivazione di questo accorgimento è che minore è la frequenza di valutazione delle azioni da parte della *policy* e minore è il numero di valutazioni delle azioni di controllo necessarie a completare un episodio. La durata temporale di un episodio dipende dal particolare problema di controllo in esame; nel caso del quadricottero è necessario che ognuno di questi abbia una durata temporale non inferiore a 40 s; non avrebbe senso considerare episodi di durata inferiore, correndo il rischio che il sistema dinamico sia troppo lento per compiere l'obiettivo impostato in un tempo più breve. Ogni valutazione delle azioni corrisponde ad un dato che viene memorizzato e richiamato dall'algoritmo per calcolare l'aggiornamento dei parametri: un numero minore di valutazione delle azioni in un episodio comporta un minor numero di dati da dover processare e di conseguenza un tempo di esecuzione più breve. Di contro si evidenzia che se si imposta ad un valore troppo piccolo la frequenza di valutazione delle azioni della *policy* dato lo stato, si incorre nella situazione in cui la legge di controllo non è in grado di identificare le caratteristiche dinamiche più veloci del mezzo. E' pertanto necessario selezionare il valore della frequenza di valutazione del comando compiendo

un compromesso tra quantità di dati da processare attraverso l'algoritmo e velocità di risposta della legge di controllo a variazioni dello stato del velivolo.

Capitolo 4

Definizione delle leggi di controllo

Nel capitolo è trattata l'impostazione del problema di RL finalizzato alla realizzazione di tre leggi di controllo per realizzare diverse modalità di volo. In primo luogo si procede all'implementazione di una legge di controllo di posizione: l'agente riceve in ingresso lo stato del modello ed elabora le azioni; tale legge di controllo ha l'obiettivo di posizionare il mezzo in un punto stabilito a partire da una condizione iniziale qualsiasi e in presenza di disturbi casuali. In [4] è proposto un algoritmo di RL per l'ottenimento di una legge di controllo di posizione implementata attraverso una rete neurale; obiettivo del lavoro è dimostrare che questa tecnica consente di ottenere un controllore in grado di stabilizzare il velivolo nel *waypoint* anche a partire da una condizione iniziale estrema, ad esempio, velivolo in caduta libera.
La seconda legge realizza un controllo di tipo vettoriale [6]: l'agente riceve in ingresso lo stato del mezzo ed elabora le azioni necessarie ad inseguire dei riferimenti in termini di velocità e angolo di rotta; questa legge di controllo è impiegata in una struttura gerarchica a cascata della quale costituisce l'anello interno. L'anello intermedio è inserito in fase di validazione ed è costituito da una funzione lineare proporzionale, che genera i riferimenti vettoriali a partire dall'errore di posizione. Il controllore gerarchico così ottenuto ha l'obiettivo di implementare un modo di volo di navigazione da
L'ultimo controllore è quello di assetto: in questo caso l'agente elabora le azioni di controllo sulla base dell'errore sull'assetto in termini di angoli di Eulero rispetto ad un segnale di riferimento. In [24] è realizzato un controllore analogo con l'obiettivo di confrontarne le prestazioni con un regolatore lineare PID.
Le leggi di controllo, come anticipato nel capitolo 2, sono realizzate per mezzo di reti neurali; i parametri delle reti sono determinati in modo automatico attraverso l'applicazione dell'algoritmo di *Reinforcement Learning*; si fa riferimento a questo processo con la locuzione allenamento della *policy*.
Per ogni legge di controllo è presentata l'impostazione del problema con la definizione dello spazio degli stati e dello spazio delle azioni, che caratterizzano gli strati di ingresso e uscita delle reti neurali; vengono poi definite le costanti di normalizzazione degli ingressi delle reti neurali. In seguito viene riportato il valore dei parametri impostati per l'algoritmo e del numero totale di passi temporali per i quali si esegue il

processo; vengono definite le caratteristiche di un episodio di impiego del controllore: è infatti necessario suddividere l'esperienza raccolta al fine di eseguire il calcolo dell'aggiornamento dei parametri. Con il termine episodio si fa dunque riferimento ad una serie di passi temporali consecutivi in cui la *policy* controlla il mezzo a partire dalla condizione iniziale al fine di raggiungere l'obiettivo assegnato. Viene definita la funzione che ad ogni istante temporale valuta il guadagno in base allo stato del modello: tale definizione è fondamentale per la corretta applicazione dell'algoritmo in quanto rappresenta l'informazione sulle prestazioni della legge di controllo; sulla base del guadagno vengono infatti eseguiti gli aggiornamenti dei parametri delle reti neurali. Infine si riporta il risultato dell'allenamento, mostrando l'andamento della somma dei guadagni ottenuti in ogni episodio nel corso del processo di definizione dei parametri.

4.1 Controllore di posizione

4.1.1 Impostazione del problema

Si allena un agente a controllare il quadrirotore a partire da una condizione iniziale qualsiasi con l'obiettivo di mantenere la posizione in un *waypoint*.
La tipologia di rete usata sia per l'*attore* che per il *critico* è il *MultiLayer perceptron* con uno strato di input, 2 strati intermedi da 64 nodi ciascuno e uno strato di output; la funzione di attivazione dei nodi è la tangente iperbolica. In tabella 4.1 sono riassunte le caratteristiche delle reti.

	attore	*critico*
hidden layer 1	64 nodi	64 nodi
hidden layer 2	64 nodi	64 nodi
uscita	4 nodi	1 nodo
attivazione	$tanh()$	$tanh()$

Tabella 4.1. Struttura dell'agente

L'agente riceve in input lo stato del quadricottero costituito da un vettore di 13 componenti:

$$\boldsymbol{Obs} = [\boldsymbol{V}_B, \boldsymbol{\omega}, \boldsymbol{Q}, \boldsymbol{Error}_P] \tag{4.1}$$

dove $\boldsymbol{V}_B, = [u, v, w]^T$ è il vettore velocità in assi corpo, $\boldsymbol{\omega} = [p, q, r]^T$ è il vettore velocità angolare in assi corpo, $\boldsymbol{Q} = [q_0, q_1, q_2, q_3]^T$ sono le componenti del quaternione di assetto e infine $\boldsymbol{Error}_P = \boldsymbol{P}_{ref} - \boldsymbol{P}$ in cui $boldsymbol P_{ref} = [X_{ref}, Y_{ref}, Z_{ref}]^T$ è la posizione di riferimento in coordinate NED e $\underline{P}$ è la posizione nello stesso sistema di riferimento.

In figura 4.1 è riportato uno schema con ingressi e uscite della rete *attore*.
Per accelerare il processo di apprendimento e migliorare i risultati finali in termini di convergenza è necessario che i nodi di input ricevano dei valori compresi nell'intervallo $[-1, 1]$; pertanto si divide ogni componente del vettore in ingresso per un coefficiente

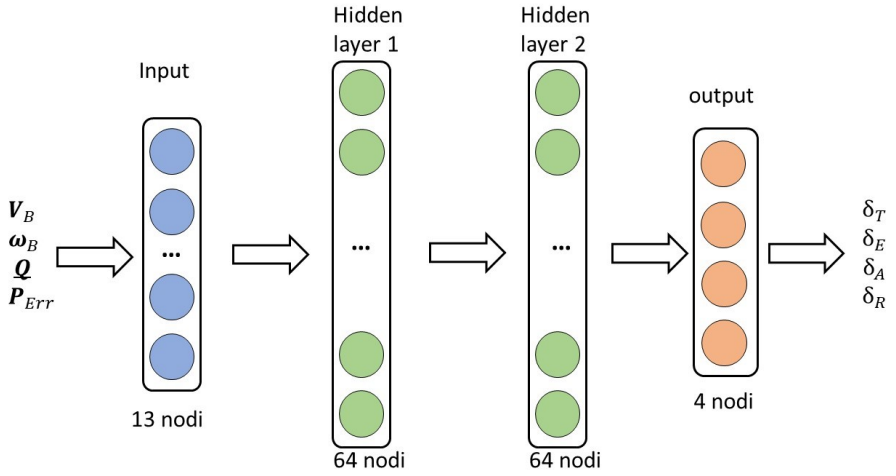

Figura 4.1. Illustrazione della rete *attore* per il controllo di posizione.

di normalizzazione che rappresenta il valore massimo accettato. In particolare la velocità massima è ±30 m/s per ogni componente, la velocità angolare massima ±50rad/s per ogni componente, il quaternione è unitario dunque non necessita di normalizzazione mentre l'errore di posizione massimo per ogni coordinata è pari a ±25 m.
Modulado il valore dei coefficienti di normalizzazione, anche a valle del processo di allenamento, si modifica l'estensione dell'intervallo atteso; tuttavia va sottolineato che i valori impostati agiscono al pari dei coefficienti di guadagno proporzionale, dunque una loro variazione eccessiva rende il sistema instabile.
La rete neurale restituisce in output un vettore di 4 componenti che sono i comandi sulla manetta media, sul momento di rollio, sul momento di beccheggio e sul momento di imbardata; tali azioni hanno valori compresi nell'intervallo $[-1, 1]$ e vengono tradotte nei 4 valori della manetta di ogni singolo motore tramite l'apposita funzione, come spiegato nella sezione 3.2. Il processo di allenamento è effettuato impiegando la *policy* sul modello che presenta contributi aerodinamici lineari rispetto a velocità e velocità angolare mentre le simulazioni di validazione a valle dell'allenamento sono effettuate sul modello che include gli effetti aerodinamici di influsso e flappeggio. La motivazione di tale scelta risiede nel fatto che, nelle prime fasi del processo di allenamento, i parametri della *policy* hanno valori casuali; la mappatura stato-azione di conseguenza è a sua volta casuale e i comandi forniti al modello comportano il raggiungimento di velocità angolari prossime ai limiti impostati: risulta complesso elaborare un modello di simulazione efficace in queste regioni dell'inviluppo di volo. E' impossibile ottenere una legge di controllo efficace senza effettuare questa semplificazione.

4.1.2 L'allenamento

Il processo è impostato creando l'agente con la funzione *PPO2()* di *Stable Baselines 2*, i valori dei parametri dell'algoritmo sono riportati in tabella 4.2.

Per ottimizzare l'uso delle risorse computazionali a disposizione occorre impostare un numero di ambienti in parallelo multiplo del numero di *thread* logici a disposizione

Parametro	Valore
N_{env}	8
N_{steps}	8192
$N_{minibatches}$	8
$N_{optEpochs}$	32
γ	0.9999
α	$2.5E-4 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$
ϵ	$0.3 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$

Tabella 4.2. Parametri dell'algoritmo

del calcolatore: in questo modo si limita la necessità di scambio di dati tra le unità di memoria dedicate dei singoli processori. I valori del rateo di apprendimento e del *clip range* decrescono linearmente durante il progresso dell'allenamento al fine di evitare modifiche eccessive dei parametri in prossimità della fine del processo. Il numero totale di passi temporali impostati come *budget* è pari a 2 milioni. L'impiego della *policy* in fase di allenamento è stocastico: i valori in uscita dalla rete neurale corrispondono ai parametri di distribuzioni Gaussiane da cui sono ricavati i comandi da applicare; questa caratteristica consente all'agente di esplorare in modo efficace le possibili mappature stato-azione.
Nella fase di valutazione a valle dell'allenamento l'impiego della rete neurale è di tipo deterministico: il valore fornito in uscita dalla *policy* corrisponde al comando effettivo applicato.

4.1.3 Caratteristiche di un episodio

Come anticipato nel capitolo 2, la raccolta dei dati su cui sono calcolati gli aggiornamenti dei parametri della rete neurale è effettuata impiegando quest'ultima direttamente sull'ambiente da controllare. Per ottimizzare il processo di allenamento si organizza l'esperienza in episodi composti da una serie consecutiva di passi temporali; al termine di ogni episodio l'algoritmo richiama automaticamente la funzione di *reset()* per ottenere nuove condizioni iniziali, mentre tra un passo temporale e il successivo viene richiamata la funzione *step()* che calcola l'evoluzione dello stato dell'ambiente data l'azione. Si precisa che la limitazione di durata temporale di un episodio è necessaria nella fase di allenamento per suddividere i dati raccolti sulle prestazioni della legge di controllo al fine di valutare l'aggiornamento dei parametri. Il passo temporale tra l'elaborazione di un'azione e la successiva vale $0.04\,s$ mentre il passo di integrazione delle equazioni di stato è di $0.01\,s$. La funzione di *reset()* inizializza lo stato con dei parametri casuali per le variabili cinematiche di posizione, velocità, velocità angolare e assetto richiamando un generatore di numeri casuali con distribuzione Gaussiana avente caratteristiche in tabella 4.3.

La posizione di riferimento è invece costante in ogni episodio di allenamento e vale: $\boldsymbol{P}_{ref} = [10,\, 15,\, -35]$ m.

Grandezza	media μ	dev.std. σ	unità
$\boldsymbol{V}_B = [u, v, w]$	0.0	0.025	[m/s]
$\boldsymbol{\omega}_B = [p, q, r]$	0.0	0.0175	[rad/s]
ϕ, θ	0.0	0.44	[rad]
X_E, Y_E	0.0	2.0	[m]
Z_E	-25.0	2.0	[m]

Tabella 4.3. Caratteristiche della distribuzione delle variabili di stato al *reset()*

Grandezza	valori limite	unità
$\boldsymbol{V}_B = [u, v, w]$	± 50	[m/s]
$\boldsymbol{\omega}_B = [p, q, r]$	± 50	[rad/s]
X_E, Y_E	± 50	[m]
Z_E	-150; 5	[m]

Tabella 4.4. Condizioni di fine episodio.

Un episodio termina al raggiungimento del tempo limite assegnato pari a 1024 passi temporali che corrispondono a 40.96 s, oppure al raggiungimento di un valore limite su uno degli stati. Tale limitazione è imposta per evitare che, al superamento dei limiti, la rete neurale operi con ingressi che eccedono l'intervallo $[-1, 1]$.
I valori terminali corrispondenti ad ogni variabile di stato sono riportati in tabella 4.4.

4.1.4 Funzione di *Reward*

La funzione che genera il segnale di guadagno ad ogni passo temporale è la seguente:

$$\begin{aligned} R = 1 \cdot q_0 &- W_{Perror}\left(\frac{|Err_Z|}{P_{max}} + \frac{|Err_X|}{P_{max}} + \frac{|Err_Y|}{P_{max}}\right) \\ &- W_{vel}\left(\frac{|u|}{V_{max}} + \frac{|v|}{V_{max}} + \frac{|w|}{V_{max}}\right) \\ &- W_{\omega}\left(\frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}}\right) \end{aligned} \tag{4.2}$$

dove q_0 è la prima componente del quaternione, i termini W_{xx} sono dei pesi adimensionali il cui valore è scelto empiricamente nell'intervallo [0, 1]; si riporta il valore di ogni peso in tabella 4.5. Err_X, Err_Y, Err_Z sono gli errori sulle tre coordinate spaziali, mentre $P_{max} = 50\,m$ è il valore di normalizzazione dell'errore di posizione; u, v, w sono le tre componenti di velocità normalizzate con $V_{max} = 50$ m/s; p, q, r sono le componenti di velocità angolare normalizzate con $\omega_{max} = 50$ rad/s.
La funzione presenta un controllo che ne limita il codominio in [0, 1]. Il valore massimo è raggiunto solo nella condizione in cui il mezzo si trova nel *waypoint* stabilito, con assetto livellato e velocità nulla. L'obiettivo dell'algoritmo è determinare i

parametri della rete neurale affinché la mappatura stato-azione risultante conduca alla massima somma dei guadagni nell'episodio; tale mappatura è raggiunta quando la rete neurale controlla il mezzo a partire dallo stato iniziale per raggiungere il punto stabilito e mantenere la posizione e l'assetto livellato il più a lungo possibile.

Grandezza	valore
W_{Perror}	0.8
W_{vel}	0.08
W_{ω}	0.1

Tabella 4.5. Pesi nel *reward*.

4.1.5 Risultato dell'allenamento

La figura 4.2 riporta l'andamento della somma dei guadagni ottenuta per un episodio in relazione al progresso dell'allenamento.

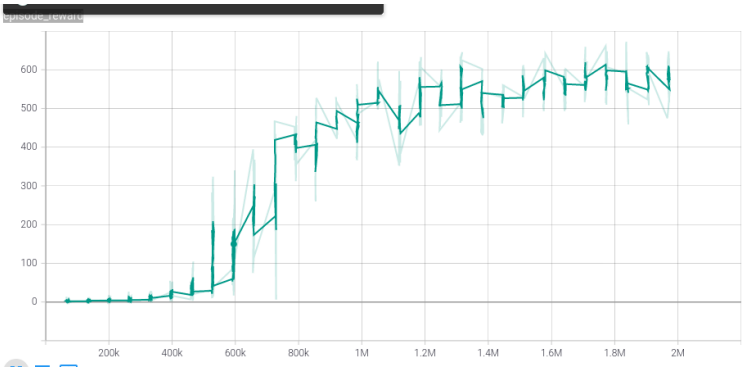

Figura 4.2. Somma dei guadagni nell'episodio durante l'allenamento.

In ascissa sono riportati i passi temporali mentre in ordinata è riportato il valore della somma dei guadagni raccolto in un episodio a seguito dei corrispondenti passi di allenamento. Il guadagno aumenta rapidamente nelle prime fasi tendendo a stabilizzarsi fino alla fine del processo, segno che l'algoritmo sta funzionando correttamente. La mappatura ottenuta è tale da annullare l'errore di posizione, la velocità, la velocità angolare e gli angoli di assetto a stazionario.
Al termine del processo vengono salvate due reti neurali:

- *Last policy*: rappresenta la *policy* finale raggiunta a valle di tutti gli aggiornamenti eseguiti fino al termine dei passi temporali impostati come *budget* per l'allenamento;
- *Best policy*: è la *policy* che durante il processo ha ottenuto la massima somma dei guadagni in un episodio; non necessariamente coincide con la *Last policy*.

Il salvataggio della *Best policy* avviene richiamando la funzione *EvalCallback()* durante l'allenamento ogni 65 536 passi temporali; in modo da effettuare una nuova

valutazione solo a valle di un aggiornamento dei parametri. La funzione esegue una simulazione deterministica della *policy* ottenuta al momento della chiamata, memorizzandola solo se ottiene una somma dei guadagni maggiore rispetto alla *best-policy* già memorizzata.
L'allenamento è stato effettuato su un calcolatore con processore *Intel Core* i7 con 8 core fisici; la procedura ha richiesto 40 minuti.

4.2 Controllore per la navigazione vettoriale

4.2.1 Impostazione del problema

Si allena una rete neurale a controllare il mezzo a partire da una condizione iniziale qualsiasi con l'obiettivo di raggiungere un riferimento sui valori della velocità in assi NED e sull'angolo di rotta ψ. Al pari del controllore del capitolo precedente, la tipologia di rete usata sia per l'*attore* che per il *critico* è il *MultiLayer perceptron* con uno strato di input, 2 strati intermedi da 64 nodi ciascuno e uno strato di output; la funzione di attivazione dei nodi è la tangente iperbolica. In tabella 4.6 sono riassunte le caratteristiche dell'agente.

	attore	*critico*
hidden layer 1	64 nodi	64 nodi
hidden layer 2	64 nodi	64 nodi
uscita	4 nodi	1 nodo
attivazione	$tanh()$	$tanh()$

Tabella 4.6. Struttura dell'agente

L'agente riceve in input un vettore di 16 componenti cosi composto:

$$\boldsymbol{Obs} = [\boldsymbol{Error}_{V\,NED},\, q3_{err},\, \boldsymbol{\omega},\, \boldsymbol{L}_{EB}] \tag{4.3}$$

dove

$$\boldsymbol{Error}_{V\,NED} = [Err_{V\,N},\, Err_{V\,E},\, Err_{V\,D}] = \boldsymbol{V}_{NEDref} - \boldsymbol{L}_{EB} \cdot \boldsymbol{V}_B \tag{4.4}$$

è l'errore sulle tre componenti di velocità in assi NED, $\boldsymbol{\omega}$ è la velocità angolare in assi corpo, $q3_{err}$ è la quarta componente del quaternione errore. Per l'assetto, la *policy* riceve in input l'*array* $\underline{L}_{EB}$ costituito da tutte le 9 componenti della matrice di rotazione dalla terna Corpo alla terna NED riportata nelle equazioni 3.7 e 3.8.
Fornire tutte le componenti del tensore costituisce una forte ridondanza in quanto sono sufficienti le 4 componenti del quaternione per descrivere ogni rotazione rigida; tuttavia la relazione lineare tra il tensore di rotazione e le componenti del vettore velocità espresso nelle due terne rende più rapido ed efficace il processo l'allenamento [4].

In figura 4.3 è riportato uno schema con ingressi e uscite della rete attore.
Il quaternione errore è valutato come:

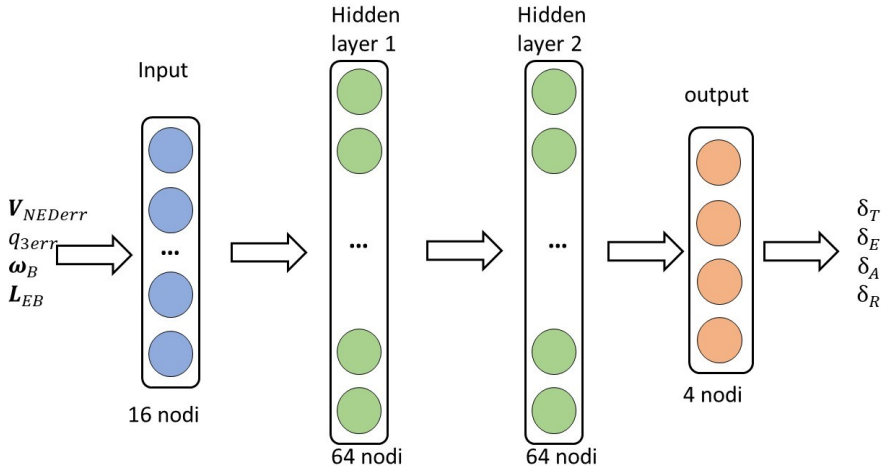

Figura 4.3. Illustrazione della rete attore per il controllo vettoriale.

$$\boldsymbol{Q}_{err} = \boldsymbol{Q}_{ref} \otimes \boldsymbol{Q}^* \tag{4.5}$$

dove il quaternione di riferimento è ottenuto imponendo il solo angolo ψ_{ref} pari all'angolo di rotta desiderato mentre $\phi_{ref} = \theta_{ref} = 0$; del quaternione errore si considera solo l'ultima componente con il seguente vincolo:

$$q3_{Err} = \begin{cases} q3_{Err} & se\ q0_{Err} \geq 0 \\ -q3_{Err} & se\ q0_{Err} < 0 \end{cases} \tag{4.6}$$

Tale vincolo viene imposto per segnalare l'errore di rotta, sempre compreso nell'intervallo $\psi_{err} \in [-\pi, \pi]$come indicato in [39]. La scelta di calcolare l'assetto di riferimento imponendo $\phi_{ref} \sqsupseteq \theta_{ref} = 0$ è fatta in quanto in condizioni stazionarie gli angoli necessari a raggiungere le velocità considerate nell'inviluppo di volo sono di piccola entità. Anche in questo caso si normalizzano gli ingressi al fine di fornire alla rete valori compresi nell'intervallo $[-1, 1]$: ogni componente del vettore in ingresso è divisa per un coefficiente che rappresenta il valore massimo accettato. In particolare per l'errore sulle componenti di velocità il valore è ± 20 m/s, per la velocità angolare ± 50 rad/s per ogni componente, la componente del quaternione errore è unitaria e non necessita di normalizzazione, come anche le componenti del tensore di rotazione. Analogamente alla funzione di controllo esposta in precedenza, anche in questo caso l'*output* è costituito dai 4 controlli su manetta media, momento di rollio, momento di beccheggio e momento di imbardata; le azioni vengono tradotte in valori di manetta per ogni motore dall'apposita funzione.

4.2.2 L'allenamento

Il *training* è impostato creando l'agente con la funzione *PPO2()*; i valori dei parametri dell'algoritmo sono:

Gli accorgimenti da tenere in considerazione per ottimizzare l'uso delle risorse computazionali sono analoghi a quelli esposti nella sezione precedente. I valori del rateo di apprendimento e del *clip range* decrescono linearmente durante il processo

Parametro	Valore
N_{env}	8
N_{steps}	8192
$N_{minibatches}$	8
$N_{optEpochs}$	64
γ	0.9999
α	$5.E-4 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$
ϵ	$0.35 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$

Tabella 4.7. Parametri dell'algoritmo

di allenamento per evitare modifiche eccessive dei pesi alla fine del processo, quando la *policy* si avvicina alla mappatura stato-azione da cui si ricava il massimo ritorno. Il numero totale di passi temporali impostati come *budget* è pari a 6 milioni.
Come per il controllore di posizione, in fase di allenamento il controllo generato dalla rete attore è di tipo stocastico mentre in fase di verifica è di tipo deterministico.

4.2.3 Caratteristiche di un episodio

L'esperienza raccolta dalla *policy* è organizzata in episodi; al termine di ogni episodio viene automaticamente richiamata la funzione di *reset()* per impostare una nuova condizione iniziale. Il valore di inizializzazione per le variabili cinematiche è casuale, selezionato a partire da distribuzioni Gaussiane con caratteristiche riportate nella tabella 4.8.

Grandezza	media μ	dev.std. σ	unità
$\underline{\omega_B} = [p, q, r]$	0.0	0.0175	[rad/s]
ϕ, θ	0.0	0.44	[rad]
ψ	0.0	3.1	[rad]

Tabella 4.8. Distribuzione delle variabili di stato al *reset()*

La velocità in assi corpo e la posizione hanno invece valore iniziale sempre nullo. I valori di riferimento per le componenti di velocità in assi NED e per l'angolo di rotta con cui è valutato il quaternione di riferimento sono impostati casualmente ad ogni esecuzione della funzione *reset()* con i parametri riportati nella tabella 4.9.

Un episodio termina al raggiungimento del limite di passi temporali pari a 1536 corrispondenti a 41.44 s di simulazione oppure al superamento di un valore limite per una delle variabili di stato; questo vincolo è motivato analogamente a quanto esposto nella sezione precedente. I valori limite corrispondenti ad ogni variabile di stato sono in tabella 4.10, se vengono raggiunti valori superiori a quelli indicati viene automaticamente richiamata la funzione *reset()*. Tale vincolo è impostato perchè al

Grandezza	media μ	dev.std. σ	unità
$V_{NORDref}$, V_{ESTref}	0.0	1.0	[m/s]
$V_{DOWNref}$	0.0	1.5	[m/s]
ψ_{ref}	0.0	3.1	[rad]

Tabella 4.9. Distribuzione delle variabili di riferimento al *reset()*

Grandezza	valori limite	unità
$\underline{V_B} = [u, v, w]$	± 20	[m/s]
$\underline{\omega_B} = [p, q, r]$	± 50	[rad/s]

Tabella 4.10. Condizioni di fine episodio.

fine di prevenire che le componenti del vettore delle osservazioni eccedano l'intervallo $[-1, 1]$.

4.2.4 Funzione di *Reward*

La funzione che genera il segnale di guadagno ad ogni passo temporale è la seguente:

$$\begin{aligned} R = 1 &- W_{Verror}\left(\frac{|Err_{Vnord}|}{V_{max}} + \frac{|Err_{Vest}|}{V_{max}} + \frac{|Err_{Vdown}|}{V_{max}}\right) \\ &- W_{\omega}\left(\frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}}\right) \\ &- W_{q3Err}|q3_{Err}| \end{aligned} \in [0,\ 1] \qquad (4.7)$$

dove i termini W_{xx} sono dei pesi reali il cui valore è scelto empiricamente, il valore di ogni peso è riportato in tabella 4.11. Err_{Vnord}, Err_{Vest}, Err_{Vdown} sono gli errori sulle tre componenti di velocità in assi NED, mentre $V_{max} = 20$ m è il valore di normalizzazione dell'errore di velocità; p, q, r sono le componenti di velocità angolare normalizzate con $\omega_{max} = 50$ rad/s. Il termine $q3_{Err}$ è l'ultima componente del quaternione di errore.
Il codominio della funzione è limitato all'intervallo $[0, 1]$. Il valore massimo è raggiunto solo quando il quadricottero raggiunge una velocità in assi terra pari al riferimento, mantenendo l'assetto costante e orientando l'asse X_B secondo l'angolo di rotta desiderato.
Durante l'allenamento i pesi della rete vengono modificati tramite l'algoritmo al fine di ottenere la mappatura stato-azione che comporta la massima somma dei guadagni nell'episodio.
I pesi nel *reward* sono riportati nella 4.11:

4.2.5 Risultato dell'allenamento

La figura 4.4 riporta l'andamento della somma dei guadagni ottenuto per un episodio nel corso dell'allenamento.

In ascissa sono riportati i passi temporali mentre in ordinata è riportato il valore della somma dei guadagni raccolto in un episodio a seguito dei corrispondenti passi

Grandezza	valore
W_{Verror}	0.8
W_{ω}	0.7
W_{q3Err}	0.9

Tabella 4.11. Pesi nel *reward*.

Figura 4.4. Somma dei guadagni nell'episodio durante l'allenamento.

di allenamento. La somma dei guadagni presenta un marcato incremento iniziale per poi stabilizzarsi, segno di un allenamento avvenuto con successo.
Anche in questo caso al termine del processo vengono salvate due reti neurali: la *Last policy* e la *Best policy*.
Il salvataggio della *Best policy* avviene richiamando la funzione *EvalCallback()* durante l'allenamento ogni 65 536 passi temporali. La motivazione di questo vincolo è analoga a quella riportata nella sezione precedente.
La funzione esegue una simulazione deterministica della *policy* ottenuta fino al momento della chiamata, salvandola se riesce ad ottenere somma dei guadagni maggiore della *Best Policy* memorizzata. L'allenamento è effettuato sullo stesso pc usato per il controllore di posizione; la procedura ha richiesto 1:30 ore.

4.3 Controllore di assetto

4.3.1 Impostazione del problema

Si determinano i parametri di una rete neurale che implementa un controllore di assetto, inseguendo riferimenti in termini di angoli di Eulero ϕ, θ, ψ. Anche in questo caso la tipologia di rete usata sia per l'*attore* che per il *critico* è il *MultiLayer perceptron* con uno strato di input, 2 strati intermedi da 64 nodi ciascuno e uno strato di output, la funzione di attivazione dei nodi è la tangente iperbolica; in tabella 4.12 sono riassunte le caratteristiche delle reti neurali implementate nell'agente.

Il vettore in ingresso alla rete ha la forma:

	attore	*critico*
hidden layer 1	64 nodi	64 nodi
hidden layer 2	64 nodi	64 nodi
uscita	4 nodi	1 nodo
attivazione	$tanh()$	$tanh()$

Tabella 4.12. Struttura dell'agente

$$\boldsymbol{Obs} = [\boldsymbol{Error}_{\Phi}, \boldsymbol{\omega}] \tag{4.8}$$

dove ω è il vettore velocità angolare mentre $\boldsymbol{Error}_{\Phi}$ è l'errore sui tre angoli di assetto:

$$\boldsymbol{Error}_{\Phi} = [\phi_{ref} - \phi, \theta_{ref} - \theta, \psi_{Err}] \tag{4.9}$$

I valori massimi per la normalizzazione degli ingressi sono 50 rad/s per le componenti di velocità angolare e 2π rad per l'errore sugli angoli di assetto.

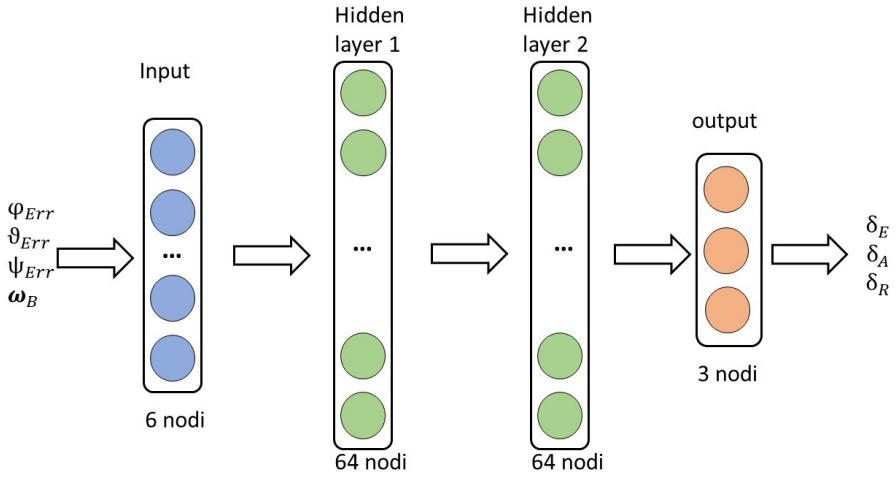

Figura 4.5. Illustrazione della rete *attore* per il controllo di assetto.

In figura 4.5 è riportato uno schema con ingressi e uscite della rete *attore*. L'errore sull'angolo di rotta è calcolato come:

$$\psi_{Err} = \begin{cases} \psi_{ref} - \psi & se \;\; \psi_{ref} - \psi \in [-\pi, \pi] \\ \psi_{ref} - \psi - 2\pi & se \;\; \psi_{ref} - \psi > \pi \\ \psi_{ref} - \psi + 2\pi & se \;\; \psi_{ref} - \psi < -\pi \end{cases} \tag{4.10}$$

Per questo controllore l'ingresso è calcolato con gli angoli di Eulero piuttosto che con il quaternione errore in quanto quest'ultimo ha dato risultati insoddisfacenti,

portando spesso il sistema in instabilità.

4.3.2 L'allenamento

Il *training* è impostato creando l'agente con la funzione *PPO2()*, i valori dei parametri dell'algoritmo sono riportati in tabella 4.13.

Parametro	Valore
N_{env}	8
N_{steps}	8192
$N_{minibatches}$	8
$N_{optEpochs}$	32
γ	0.9999
α	$6.E-5 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$
ϵ	$0.32 \cdot (1 - N_{elapsedS}/N_{TOTSteps})$

Tabella 4.13. Parametri dell'algoritmo

Anche per questo allenamento valgono gli stessi accorgimenti riportati nei due capitoli precedenti riguardo l'uso delle risorse computazionali e riguardo l'andamento del rateo di apprendimento e del *clip-range*. Il numero totale di passi temporali impostati come *budget* è pari a 4 milioni. Anche in questo caso l'allenamento è eseguito estraendo dalla rete neurale comandi di natura stocastica per poi passare a valori deterministici in fase di simulazione.
La *policy* restituisce in uscita 3 valori: le azioni sul momento di rollio, beccheggio ed imbardata. E' libero il valore della manetta media in quanto per questo modo di volo si forniscono dei riferimenti solo in termini di assetto. Questa caratteristica predispone il controllore ad interfacciarsi con altri algoritmi di controllo di livello superiore oppure con comandi forniti da un pilota umano: quest'ultimo sceglie manualmente il valore della manetta e gli angoli di riferimento e la rete neurale controlla in retroazione i momenti.

4.3.3 Caratteristiche di un episodio

Il valore di inizializzazione delle variabili cinematiche e dei riferimenti è casuale selezionato ad ogni *reset()* da distribuzioni Gaussiane aventi le caratteristiche riportate in tabella 4.14.

Gli angoli di assetto vengono inizializzati a 0.
Le condizioni terminali per l'episodio sono il raggiungimento del tempo limite impostato in 1024 passi temporali ovvero 40.96 s, oppure il superamento dei valori limite sulle componenti di velocità angolare $\boldsymbol{\omega}_B$ pari a 50 rad/s.

Grandezza	media μ	dev.std. σ	unità
$\underline{V_B} = [u, v, w]$	0.0	0.025	$[m/s]$
$\underline{\omega_B} = [p, q, r]$	0.0	0.0175	$[rad/s]$
ϕ_{ref}, θ_{ref}	0.0	0.35	$[rad]$
ψ_{ref}	0.0	3.1	$[rad]$

Tabella 4.14. Caratteristiche della distribuzione delle variabili di stato e dei riferimenti al *reset()*

4.3.4 Funzione di *Reward*

Il guadagno ad ogni passo è calcolato tramite la funzione:

$$R = 1 - W_{\phi-\theta Err}\left(\frac{|Err_\phi|}{\phi_{max}} + \frac{|Err_\theta|}{\theta_{max}}\right) - W_{\psi Err}\frac{|Err_\psi|}{\psi_{max}} \in [0,\ 1] \qquad (4.11)$$
$$-W_\omega\left(\frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}} + \frac{|p|}{\omega_{max}}\right)$$

dove i pesi W_{xx} sono riportati in tabella 4.15; $\phi_{max} = \theta_{max} = 2 * \pi$ rad e $\psi_{max} = 1.5\pi$ rad sono i valori massimi per gli errori sui tre angoli di Eulero Err_ϕ, Err_θ, Err_ψ; p, q, r sono le tre componenti di velocità angolare per cui il valore massimo vale $\omega_{max} = 50$ rad/s.
Il valore massimo del guadagno è raggiunto in tutti i passi temporali in cui l'errore rispetto agli angoli di riferimento è nullo insieme alla velocità angolare; questa caratteristica ha lo scopo di allenare la *policy* a raggiungere gli angoli di riferimento nel più breve tempo possibile.

Grandezza	valore
$W_{\phi-\theta Err}$	1.0
$W_{\psi Err}$	0.9
W_ω	0.9

Tabella 4.15. Pesi nel *reward*.

4.3.5 Risultato dell'allenamento

La figura 4.6 riporta l'andamento della somma dei guadagni in ordinata mentre in ascissa sono i passi temporali.

Anche per questo agente è positivo il fatto che la somma dei guadagni ottenuti in un episodio si stabilizza e mantiene costante dopo circa 2 milioni di passi di allenamento, segno del fatto che la *policy* converge verso la mappatura stato-azione desiderata.
E' salvata sia l'ultima *policy* ottenuta che quella che durante il training ha ottenuto il massimo valore della somma dei guadagni in simulazioni deterministiche. Il comando utilizzato per valutare l'agente ad intervalli di 65 536 passi è anche in questo caso

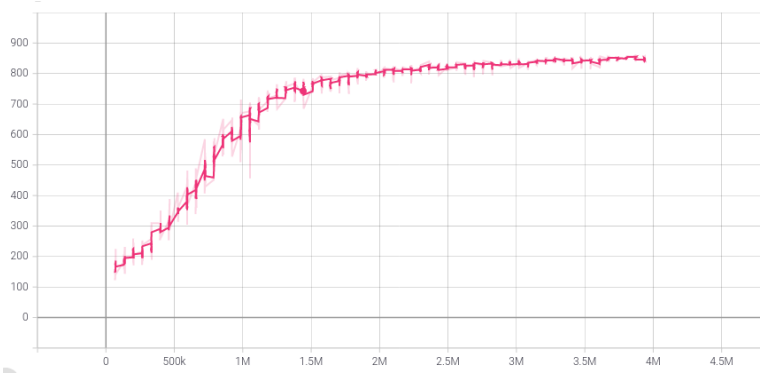

Figura 4.6. Somma dei guadagni nell'episodio durante l'allenamento.

EvalCallback(); l'obiettivo dell'imposizione di questo vincolo temporale è analogo a quello spiegato per il controllore di posizione. L'allenamento è effettuato sullo stesso computer usato per gli altri due controllori e ha richiesto 1 ora.

Capitolo 5

Risultati

Si riportano in questo capitolo i risultati ottenuti in simulazione con leggi di controllo basate sugli agenti illustrati nei capitoli precedenti. Dato che le modalità di volo realizzate da ogni agente presentano tra loro caratteristiche differenti, risulta complesso individuare una modalità di simulazione unica.
Il primo controllore viene testato in campagna Monte Carlo, il secondo in una simulazione di missione a *waypoints* mentre il terzo agente viene testato esaminando le risposte a segnali in ingresso a gradino sui tre angoli di riferimento.

5.1 Controllore di posizione

Si riportano i risultati di una serie di 25 simulazioni in cui il quadricottero viene inizializzato con uno stato sempre diverso in termini di posizione, velocità , velocità angolare e assetto. L'obiettivo è raggiungere e mantenere una posizione specifica. L'architettura del sistema è in figura 5.1.

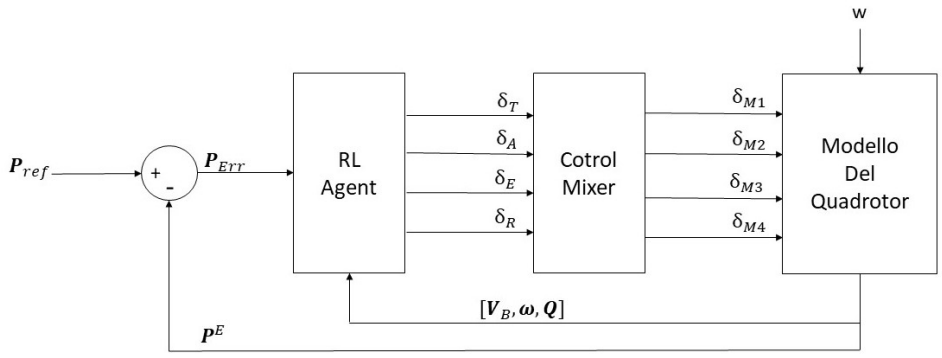

Figura 5.1. Architettura del controllore di posizione.

In accordo con il metodo di Monte Carlo le condizioni iniziali cinematiche sono selezionate ad ogni *reset()* da distribuzioni Gaussiane aventi le caratteristiche in tabella 5.1.

Grandezza	media μ	dev.std. σ	unità
$\underline{V_B} = [u, v, w]$	0.0	1.5	[m/s]
$\underline{\omega_B} = [p, q, r]$	0.0	0.035	[rad/s]
ϕ, θ	0.0	0.44	$[rad]$

Tabella 5.1. Caratteristiche della distribuzione delle variabili di stato al *reset()*.

La posizione è inizializzata in punti casuali su una circonferenza centrata nell'origine con raggio pari a 10 m posta ad una quota di 2 m. Il *waypoint* da raggiungere si trova nel punto $[0, 0, 12]$ [m].

Nella tabella che segue si riportano i valori iniziali delle variabili cinematiche per 10 delle 25 simulazioni effettuate, selezionate a campione:

	$\boldsymbol{V}_{Bini}$ [m/s]	$\boldsymbol{\omega}_{Bini} \cdot 10^{-2}$ [rad/s]
Sim 1	$[1.65, -1.01, -0.04]$	$[-0.0730, 3.08, 3.79]$
Sim 2	$[-0.97, 2.07, -0.48]$	$[1.14, 1.84, -0.719]$
Sim 3	$[-0.04, 1.55, -1.73]$	$[-1.80, 3.78, 1.00]$
Sim 4	$[1.89, -2.08, 1.83]$	$[-0.229, 3.87, 2.60]$
Sim 5	$[-0.25, -0.04, -1.72]$	$[-3.03, -2.88, 0.517]$
Sim 6	$[5.59, -1.54, 1.04]$	$[-2.66, -1.99, 2.60]$
Sim 7	$[1.23, -0.42, -0.23]$	$[-1.63, -1.78, -1.52]$
Sim 8	$[-0.06, -0.28, -0.70]$	$[-0.142, 4.42, 2.09]$
Sim 9	$[0.53, 0.71, -1.36]$	$[4.08, -0.0173, -2.84]$
Sim 10	$[0.76, -1.45, -0.49]$	$[-0.0985, 2.26, 1.27]$

Tabella 5.2. Valori iniziali delle variabili cinematiche.

Il sistema di controllo deve gestire delle condizioni iniziali non nulle; inoltre sono inseriti dei disturbi in termini di accelerazioni lineari ed angolari caratterizzati da distribuzioni Gaussiane (tabella 5.3), che simulano turbolenze aerodinamiche casuali. Questi contributi sono inseriti per dimostrare che il controllore ottenuto è efficace nel controllare il quadrirotore in presenza di disturbi casuali.

Grandezza	media μ	varianza σ^2	unità
$\boldsymbol{V}_B = [u, v, w]$	0.0	0.01	[m/s^2]
$\boldsymbol{\omega}_B = [p, q, r]$	0.0	0.00175	[rad/s^2]

Tabella 5.3. Caratteristiche delle distribuzioni delle azioni di disturbo.

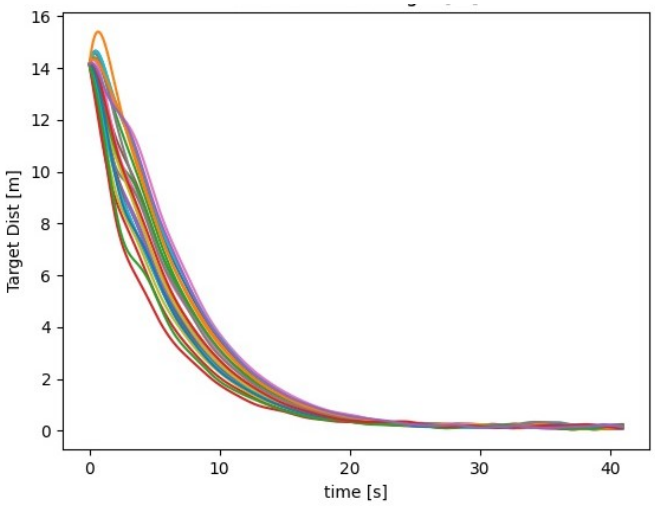

Figura 5.2. Distanza in metri dal *waypoint* durante le simulazioni.

In figura 5.2 si osserva come in ogni simulazione il controllore riesca in tempi brevi ad annullare la distanza dal *target*, annullando correttamente l'errore sulle tre coordinate spaziali e infine stabilizzando il quadricottero nel punto stabilito per mantenere la posizione il più a lungo possibile.

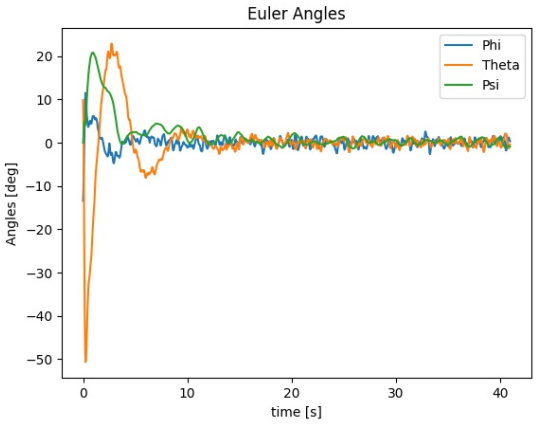

Figura 5.3. Andamento degli angoli di Eulero durante una simulazione.

In figura 5.3 si osserva la compensazione dei disturbi tramite l'andamento degli angoli di Eulero nel corso di una delle 25 simulazioni (l'andamento è paragonabile in tutte le altre prove). In particolare si nota un'importante perturbazione dell'assetto legata alla condizione iniziale; in seguito si osserva un annullamento degli angoli di assetto con sovrapposizione dell'effetto delle accelerazioni di disturbo. Si sottolinea che il valore finale dell'angolo di rotta pari a 0^o rappresenta un vincolo legato al fatto che l'obiettivo di questa *policy* è quello di posizionare il quad in un punto stabilito a partire da qualsiasi condizione iniziale, non la realizzazione di un controllo per la navigazione vettoriale.

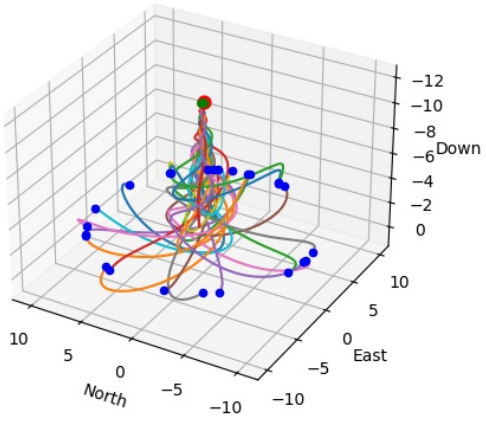

Figura 5.4. Traiettoria del centro di massa nelle diverse simulazioni.

La figura 5.4 riporta la traiettoria descritta dal centro di massa in tutte le simulazioni; i punti di colore blu rappresentano la posizione iniziale del quad, i punti di colore verde rappresentano le posizioni finali mentre il punto rosso è il target. Il punto di arrivo è sostanzialmente sempre coincidente con il *waypoint* e le traiettorie descritte dimostrano che il controllore ottenuto è efficace nell'annullare l'errore di posizione in ogni simulazione.
I parametri statistici di media e varianza dell'errore finale di posizione per le simulazioni sono in tabella 5.4. L'istogramma di figura 5.5 riporta graficamente la distribuzione della distanza finale dal *waypoint* nelle simulazioni.

media	$0.165\,m$
varianza	0.0024

Tabella 5.4. Media e varianza della distanza finale dal *waypoint*.

5.1.1 Test Monte Carlo con variazione dei parametri del modello

Al fine di provare la robustezza della *policy* allenata nel posizionare il mezzo, è stato ripetuto il test Monte Carlo su un modello di quadricottero avente parametri costruttivi diversi rispetto a quelli del *Hummingbird*. In particolare sono stati assegnati valori relativi ad un mezzo più grande e pesante; la tabella 5.5 riporta un confronto dei parametri tra il modello simulato nel test descritto nella sezione precedente e quello con parametri modificati.
I risultati della simulazione in termini di distanza dal *target* e traiettoria sono in fig. 5.6 e 5.7.

I parametri statistici della distanza finale dal punto obiettivo sono confrontati con quelli della prova sul *Hummingbird* nella tabella 5.6.

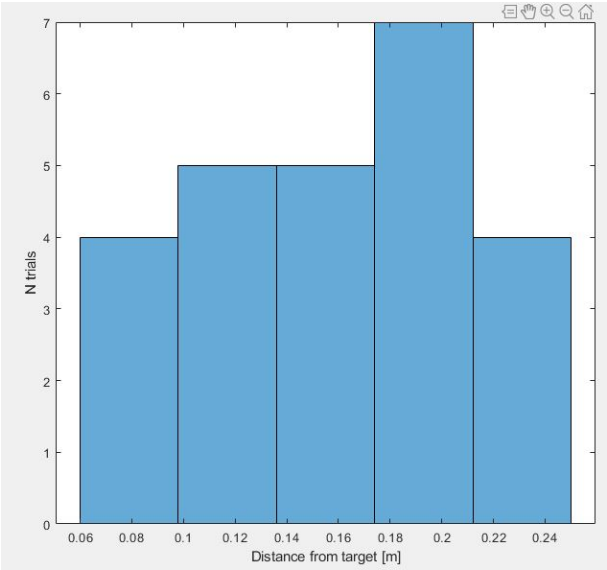

Figura 5.5. Istogramma della distanza finale dal *waypoint*.

Parametro	val. *Hummingbird*	val. modificato	unità
L_X	0.34	0.75	[m]
L_y	0.34	0.75	[m]
massa motore	0.04	0.2	[kg]
massa corpo	0.484	0.70	[kg]
massa batteria	0.186	0.40	[kg]
massa totale	0.71	1.9	[kg]
D elica	0.2032	0.381	[m]
K_t elica	$2.27e-4$	$2.194e-3$	$[\mathrm{N\,s^2}]$
K_q elica	$3.22e-6$	$5.9e-5$	$[\mathrm{N\,m\,s^2}]$

Tabella 5.5. Variazione dei parametri del modello.

	Hummingbird	parametri modificati
media	0.165 m	0.1697 m
varianza	0.0024	0.0034

Tabella 5.6. Media e varianza della distanza finale dal *target* del *Hummingbird* e del modello con parametri modificati.

La figura 5.8 riporta la distribuzione statistica dell'errore finale sia nel caso del *Hummingbird* che nel caso di modello con parametri modificati. Le figure 5.6 e 5.7 riportano rispettivamente la distanza dal *waypoint* e la traiettoria del baricentro del

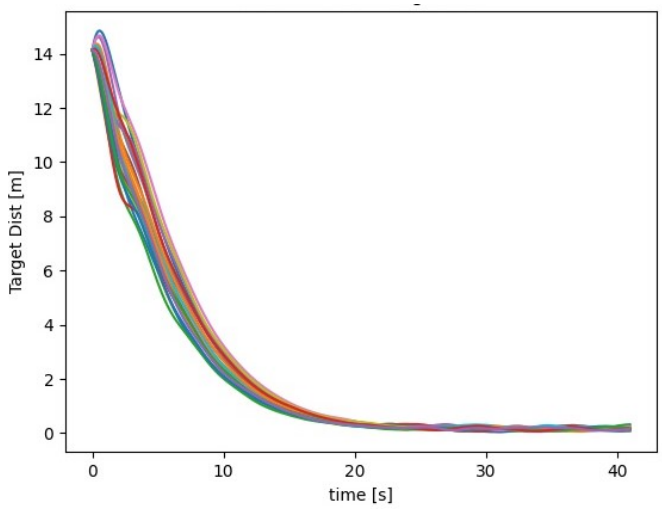

Figura 5.6. Distanza dal *target* del modello con parametri modificati.

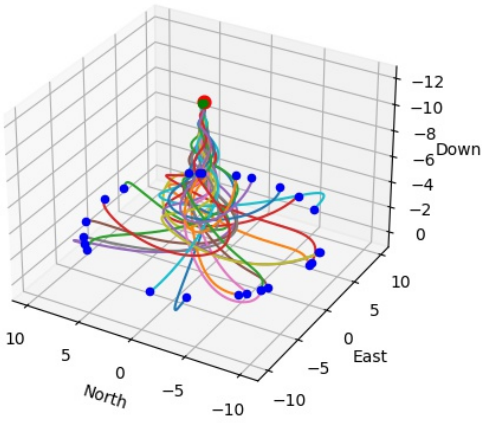

Figura 5.7. Traiettorie del modello con parametri modificati.

mezzo con parametri modificati nel corso delle simulazioni. Si osserva una media dell'errore simile nel controllo dei due quadricotteri ma una deviazione standard maggiore nel caso con parametri modificati. Si conclude che la *policy* allenata è in grado di gestire incertezze parametriche non piccole, seppur con prestazioni leggermente peggiori in termini di errore a stazionario.

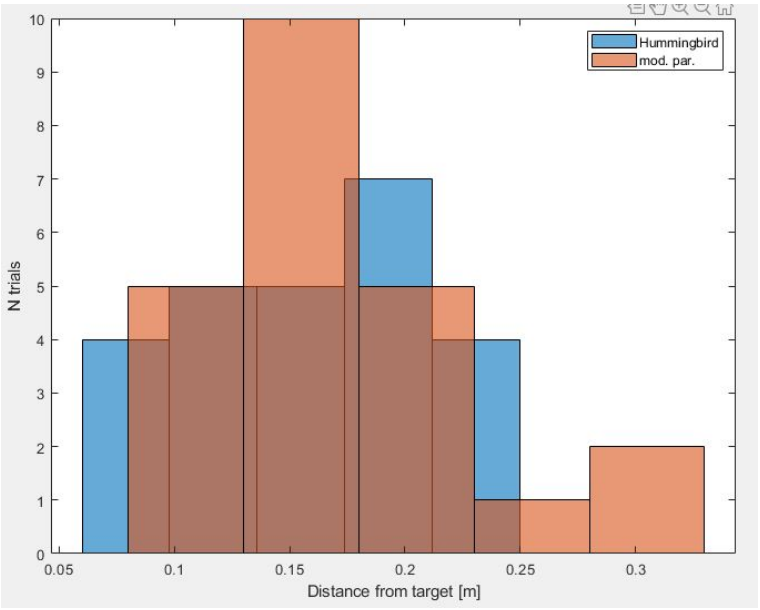

Figura 5.8. Istogramm della distanza finale dal *waypoint* per *Hummingbird* e modello con parametri modificati a confronto.

5.2 Controllore per la navigazione vettoriale

Per validare la *policy* per la navigazione vettoriale si effettua la simulazione di una missione di volo. Si impostano 6 *waypoints* da raggiungere consecutivamente a partire da una condizione iniziale di volo livellato, simulando una missione di esplorazione. Il riferimento in termini di velocità e angolo di rotta è assegnato da un controllore proporzionale a partire dall'errore di posizione. La posizione iniziale è $\boldsymbol{P}_{ini}^{E} = [0,\, 0,\, -5]\, m$, mentre le coordinate dei *waypoints* in sequenza sono in tabella 5.7.
In tabella è riportato anche l'istante temporale in cui avviene il passaggio del riferimento da un *wypoint* al successivo; il modello dunque deve raggiungere e mantenere la posizione fino al passaggio del riferimento al *waypoint* successivo trascorso il tempo stabilito. I tempi riportati in tabella 5.7 sono scelti per consentire al modello di raggiungere un punto di rotta e stabilizzarsi prima di passare al successivo.

La figura 5.9 illustra la sequenza dei *waypoints* impostati. I disturbi sono rappresentati da accelerazioni lineari ed angolari con media nulla e distribuzione Gaussiana con le caratteristiche mostrate nella tabella 5.8.

5.2.1 Architettura del controllore

La *policy* in esame è pensata per operare in una struttura gerarchica, come anello di controllo interno; per realizzare il controllo di posizione è interfacciata con un *loop* intermedio che genera i riferimenti per la velocità in assi NED tramite un'azione proporzionale all'errore di posizione. La gerarchia degli anelli di controllo è ripresa dal lavoro [16]. La legge di controllo di posizione è la seguente:

	$\boldsymbol{P}^E_{WP}$ [m]	tempo di impostazione del WP [s]
WP 1	[0, 0, −20]	14
WP 2	[15, 0, −20]	30
WP 3	[15, 15, −20]	48
WP 4	[0, 15, −20]	64
WP 5	[0, 0, −20]	80
WP 6	[0, 0, −5]	100

Tabella 5.7. Coordinate dei punti rotta con l'istante temporale in cui viene impostato il *waypoint*.

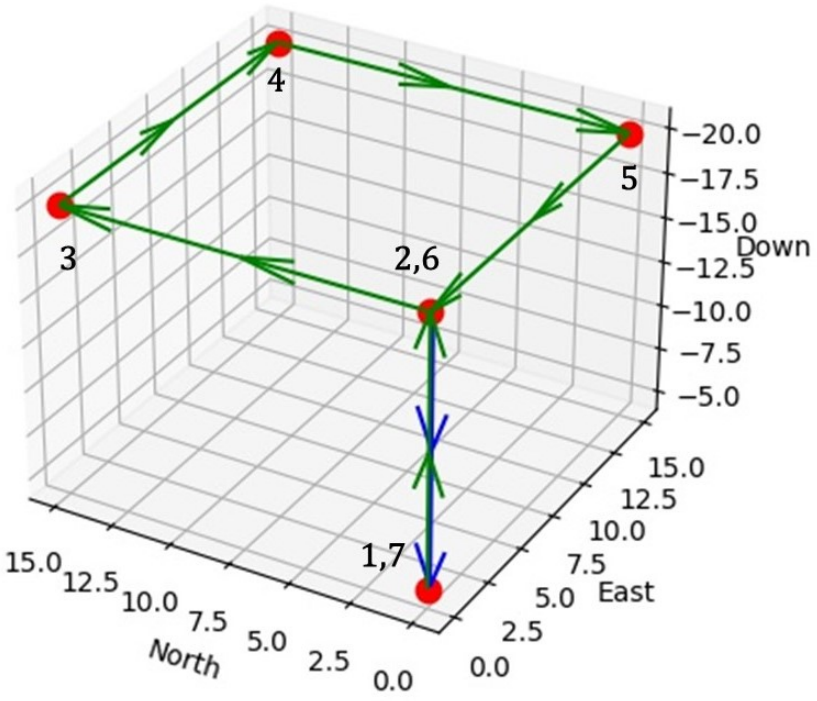

Figura 5.9. Traiettoria della missione.

	media μ	dev.std. σ	unità
$\dot{\boldsymbol{V}}_{\boldsymbol{B}D} = [\dot{u}, \dot{v}, \dot{w}]$	0.0	0.005	[m/s^2]
$\dot{\boldsymbol{\omega}}_D = [\dot{p}, \dot{q}, \dot{r}]$	0.0	0.175	rad/s^2]

Tabella 5.8. Parametri statistici delle distribuzioni Gaussiane dei disturbi casuali.

$$\begin{cases} \dot{X}^E_{ref} = V_{NORDref} = K_p\,(X^E_{ref} - X^E) \\ \dot{Y}^E_{ref} = V_{EASTref} = K_p\,(Y^E_{ref} - Y^E) \\ \dot{Z}^E_{ref} = V_{DOWNref} = K_p\,(Z^E_{ref} - Z^E) \end{cases} \tag{5.1}$$

dove $[\dot{X}^E_{ref}, \dot{Y}^E_{ref}, \dot{Z}^E_{ref}]$ sono le tre componenti della velocità di riferimento in assi NED, $[(X^E_{ref} - X^E), (Y^E_{ref} - Y^E), (Z^E_{ref} - Z^E)]$ sono le tre componenti dell'errore di posizione nello stesso sistema di riferimento mentre K_p è il guadagno proporzionale del controllore lineare che elabora i riferimenti vettoriali per la *policy* in base all'errore di posizione.
Il valore di ogni componente del riferimento sulla velocità in assi terra è limitato a ± 2 m/s per evitare comandi troppo bruschi. La *policy* riceve tra le osservazioni, le componenti dell'errore di velocità in assi terra calcolato secondo la 4.4.
L'errore sull'angolo di rotta è calcolato tramite il quaternione errore come nelle 4.5 e 4.6, impostando l'angolo di rotta di riferimento come:

$$\psi_{ref} = atan2\left(\frac{Y_{err}}{X_{err}}\right) = \begin{cases} atan2\left(\frac{Y_{err}}{X_{err}}\right), & se \quad X_{err} > 0 \\ atan2\left(\frac{Y_{err}}{X_{err}}\right) + \pi, & se \quad X_{err} < 0 \; e \; Y_{err} \geq 0 \\ atan2\left(\frac{Y_{err}}{X_{err}}\right) - \pi, & se \quad X_{err} < 0 \; e \; Y_{err} < 0 \\ \frac{\pi}{2}, & se \quad X_{err} = 0 \; e \; Y_{err} > 0 \\ -\frac{\pi}{2}, & se \quad X_{err} = 0 \; e \; Y_{err} < 0 \\ Non\ Definito, & se \quad X_{err} = 0 \; e \; Y_{err} = 0 \end{cases} \tag{5.2}$$

con:

$$\begin{cases} X_{err} = X^E_{ref} - X^E \\ Y_{err} = Y^E_{ref} - Y^E \end{cases} \tag{5.3}$$

Per evitare il punto di singolarità $X^E_{err} = 0$ e $Y^E_{err} = 0$ e comandi di imbardata eccessivi in prossimità del *waypoint*, l'aggiornamento dell'angolo di rotta di riferimento è effettuato solo se la distanza D_{HOR} è maggiore di 1.5 m:

$$D_{HOR} = \sqrt{X^2_{err} + Y^2_{err}} \tag{5.4}$$

L'architettura del sistema è riportata in figura 5.10.

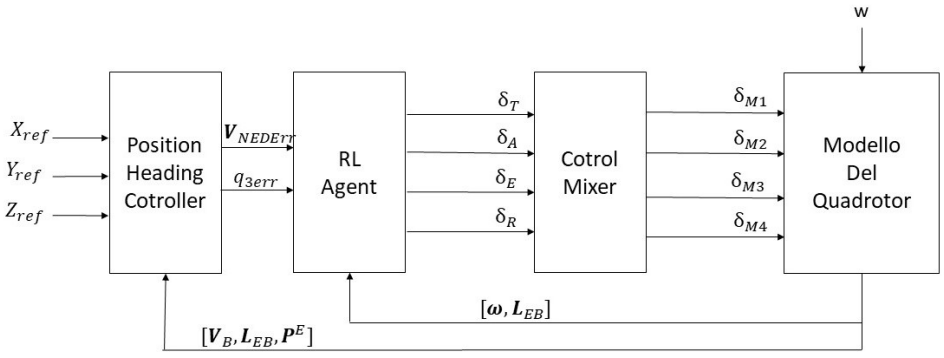

Figura 5.10. Architettura del sistema di controllo vettoriale.

5.2.2 Risultati della simulazione

La traiettoria del mezzo rispetto ai 5 *waypoints* è riportata nella figura 5.11.

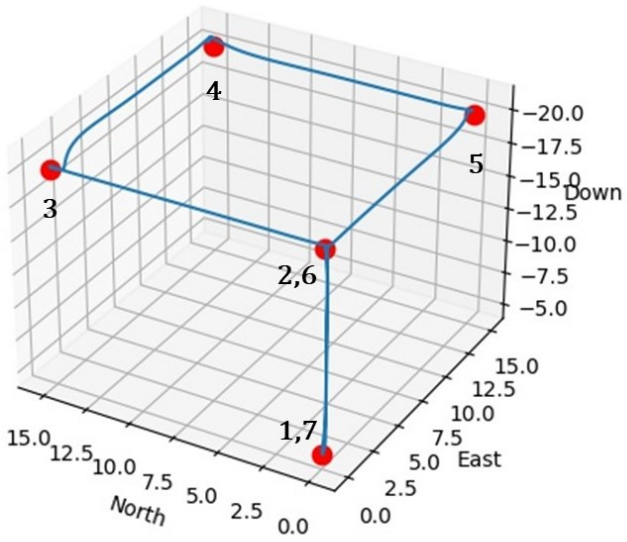

Figura 5.11. Traiettoria del quadricottero rispetto ai *waypoints*.

Nelle figure 5.12 e 5.13 si vede lo sviluppo della traiettoria a tempi diversi. I grafici dove è riportata anche la terna di assi corpo (l'asse X_B è in rosso) mostrano la posizione e l'orientamento del modello durante la simulazione. Dalle figure riferi

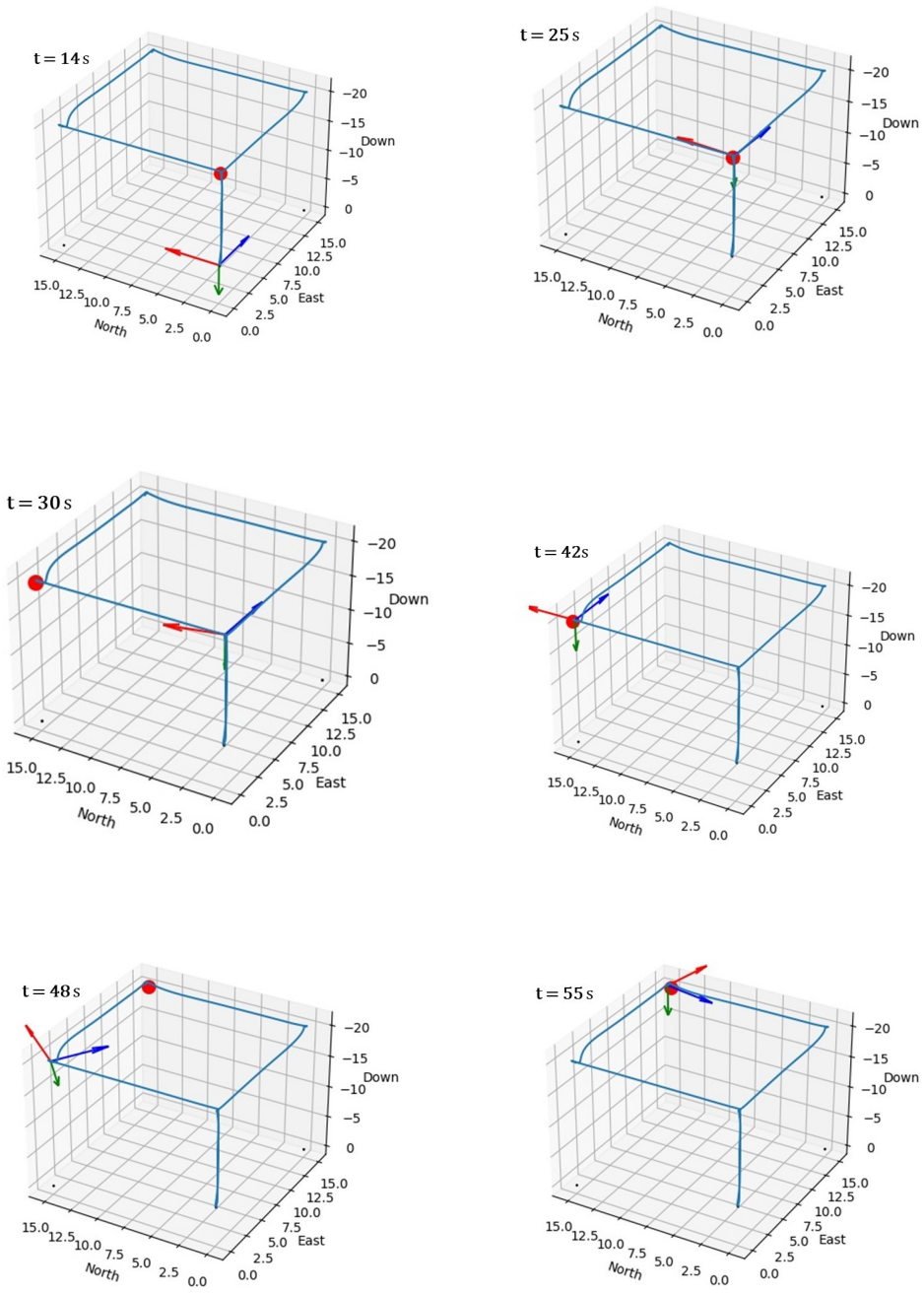

Figura 5.12. Evoluzione della traiettoria per tempi compresi nell'intervallo [0, 55] s, la posizione del quadricottero è individuata dalla terna di assi corpo.

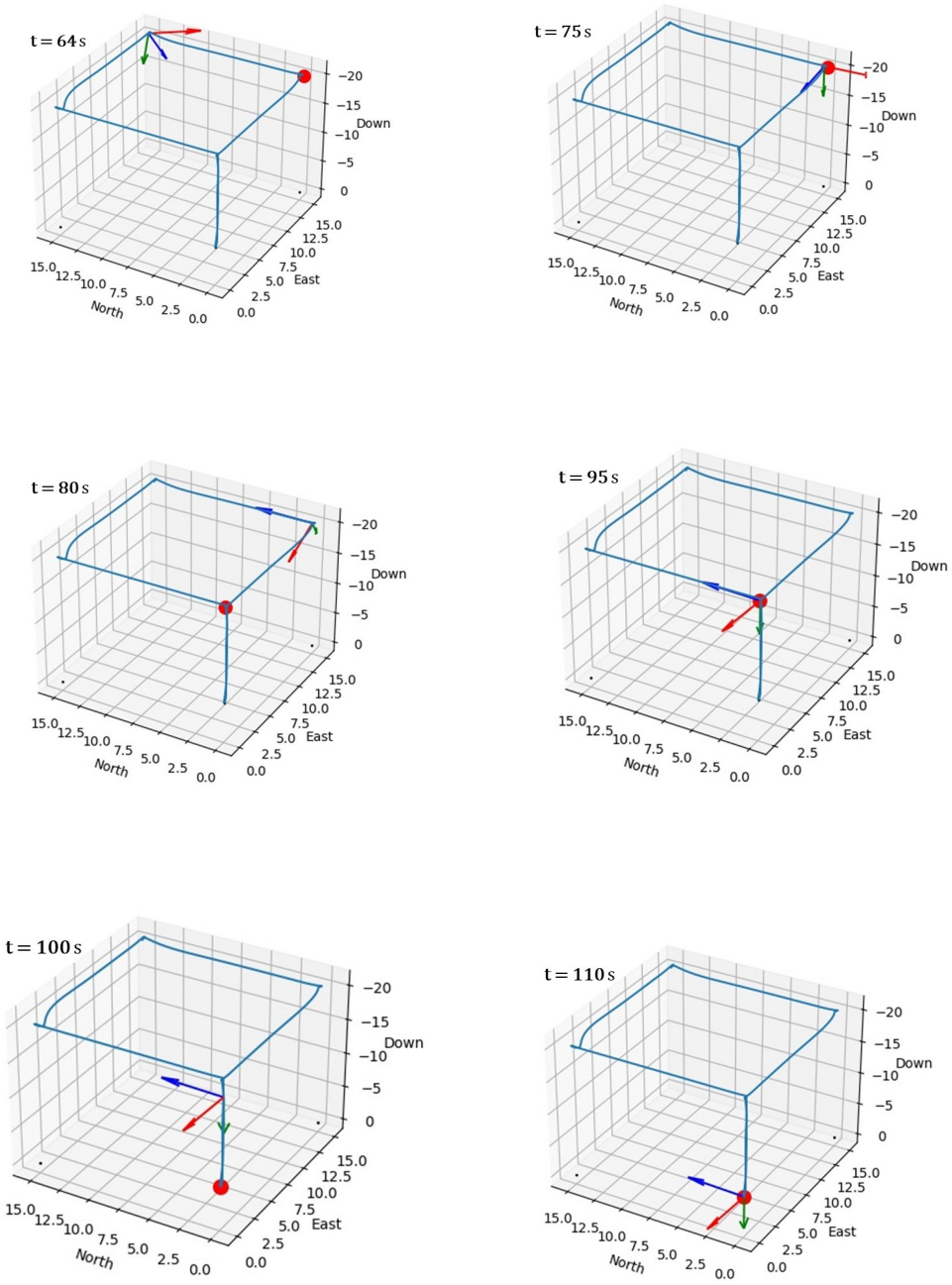

Figura 5.13. Evoluzione della traiettoria per tempi compresi nell'intervallo [55, 110] s.

Il quadricottero mantiene correttamente la posizione una volta raggiunto il riferimento.
Nelle figure 5.12 per t=30 s e t=48 s e nella figura 5.13 per t=64 s, t=80 s e t=100 s sono rappresentati gli istanti successivi alla variazione del *waypoint* di riferimento; per questi fotogrammi si osserva in particolare come l'asse X_B non punti nella direzione della velocità, tangente alla traiettoria. Questo è legato al fatto che nel modello la trazione è assunta orientata secondo l'asse Z_B, la *policy* comanda il quadrotor per ottenere l'assetto necessario a generare una componente della trazione in direzione orizzontale. Dalle figure si osserva anche come l'asse X_B sia sempre orientato nella direzione del *waypoint* di riferimento, segno del fatto che il controllore è efficace nel regolare l'angolo di rotta.

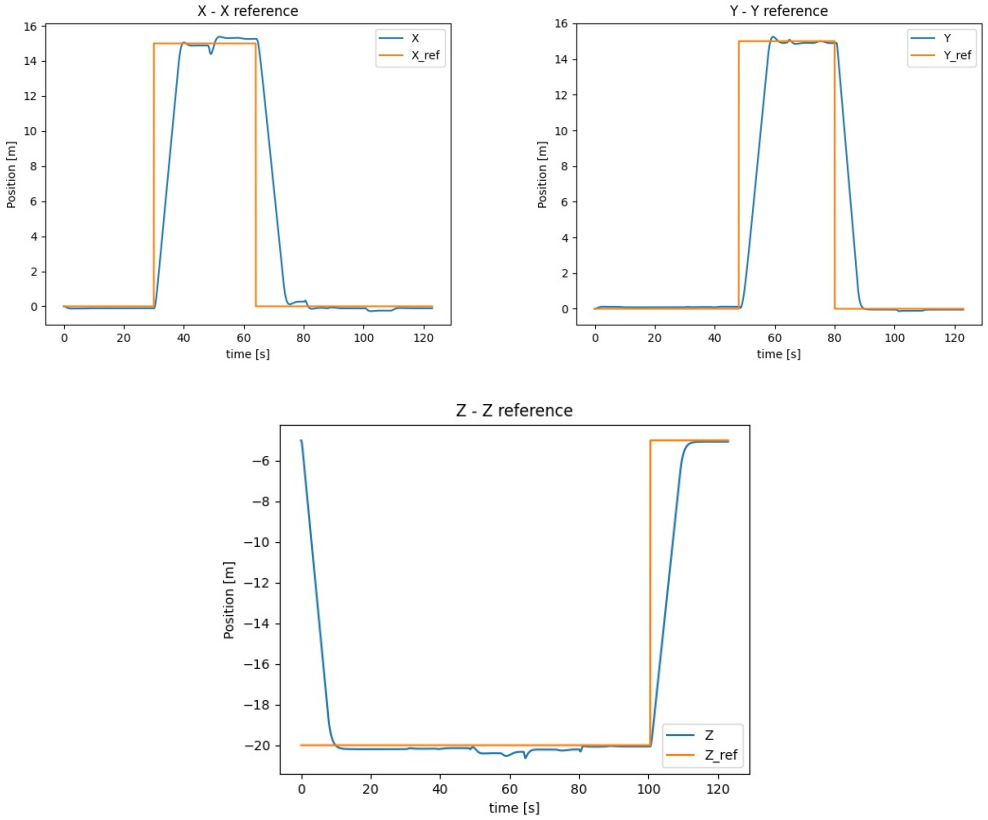

Figura 5.14. Coordinate di posizione rispetto al riferimento.

Nei grafici in figura 5.14 si osserva l'andamento della posizione del quadricottero rispetto alla posizione di riferimento.

La figura 5.15 mostra l'andamento delle azioni di controllo durante la simulazione. I picchi dei comandi sui momenti si verificano negli istanti successivi alla variazione del riferimento: il controllore agisce per raggiungere la posizione e l'angolo di rotta desiderato. I comandi sulla manetta media a inizio e fine simulazione corrispondono alla variazione della quota di riferimento.
Infine nella tabella 5.9 è riportato il valore della distanza a stazionario dai *waypoint* nel corso della missione. Si conclude che il controllore è efficace nel gestire una

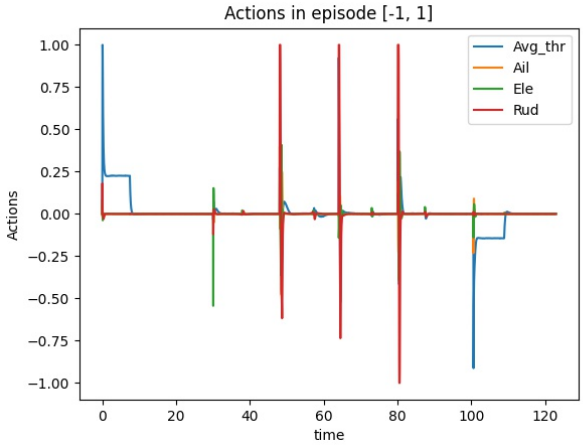

Figura 5.15. Azioni in uscita dall'agente durante la simulazione.

missione di volo a *waypoints*, compensando anche azioni di disturbo casuali.

	D_{ss} $[m]$
WP 1	0.2413
WP 2	0.2192
WP 3	0.4275
WP 4	0.3551
WP 5	0.1389
WP 6	0.1392

Tabella 5.9. Distanza a stazionario dai *waypoints*.

5.3 Simulazione del controllore di Assetto

Per validare le prestazioni del controllore di assetto vengono effettuate delle simulazioni di risposta al gradino.
Il risultato viene confrontato con la risposta di un regolatore di assetto lineare di tipo PID. La simulazione della *policy* è eseguita in ambiente *Python* mentre la simulazione del regolatore PID è eseguita in un modello *Simulink*; i risultati della prova in ambiente *Python* vengono in seguito importati in *Matlab* per elaborare i grafici di confronto delle prestazioni dei due controllori.

5.3.1 Controllore PID

Per effettuare il confronto si è sviluppato un controllore di assetto proporzionale, integrale, derivativo (PID) seguendo la trattazione in [40]. Il controllore genera comandi sui momenti con la seguente legge:

$$\begin{aligned} \delta A &= K_P(\phi_d - \phi) + K_I \int(\phi_d - \phi)dt - K_D p \\ \delta E &= K_P(\theta_d - \theta) + K_I \int(\theta_d - \theta)dt - K_D q \\ \delta R &= K_P(\psi_d - \psi) + K_I \int(\psi_d - \psi)dt - K_D r \end{aligned} \tag{5.5}$$

dove $\Phi_d = [\phi_d, \theta_d, \psi_d]$ sono i valori di riferimento degli angoli di assetto, $\Phi = [\phi, \theta, \psi]$ gli angoli misurati e p, q, r sono le tre componenti di velocità angolare in assi corpo; tali componenti costituiscono il contributo derivativo del controllore.

Il valore dei guadagni è assegnato con la procedura di *tuning* manuale sfruttando lo strumento del *Control System Designer* messo a disposizione da Simulink; nella tabella 5.10 sono riportati i valori dei guadagni e i margini di fase corrispondenti ad ogni *loop* di controllo.

	K_P	K_I	K_D	P.M. [deg]	freq. [Hz]
Rollio	0.5482	0.6481	0.109	63.7	4.02
Beccheggio	0.5482	0.6481	0.109	62.7	4.02
Imbardata	0.3961	0.04456	0.2156	54.3	1.68

Tabella 5.10. Guadagni e margini di fase del controllore di assetto.

L'architettura del regolatore è in figura 5.16.

5.3.2 Simulazioni

La condizione iniziale è di volo a punto fisso.
Vengono aggiunti dei disturbi casuali che simulano gli effetti della turbolenza aerodinamica. Per rendere attendibile il confronto dei risultati, data la difficoltà di apportare modifiche al blocco di integrazione delle accelerazioni fornito dalla *Mathworks*, i disturbi vengono aggiunti alle azioni esterne, moltiplicati per massa ed inerzia del mezzo; la relazione è la seguente:

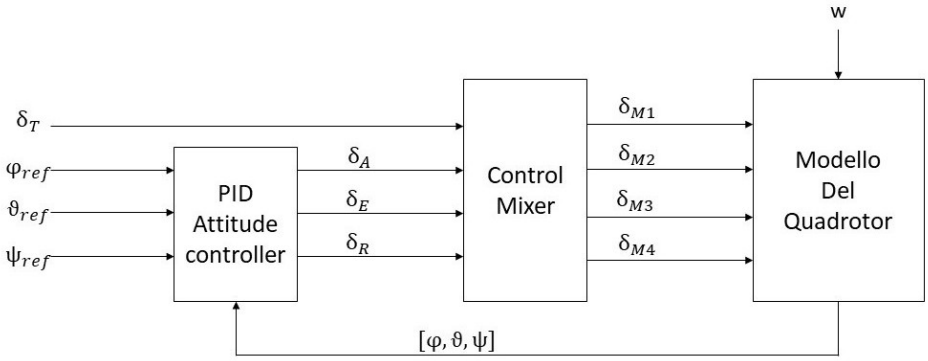

Figura 5.16. Architettura del regolatore di assetto PID.

$$\begin{aligned} \boldsymbol{F}_D &= m \cdot \dot{\boldsymbol{V}}_{\boldsymbol{B}D} \\ \boldsymbol{G}_D &= \boldsymbol{I} \cdot \dot{\boldsymbol{\omega}}_{\boldsymbol{D}} \end{aligned} \tag{5.6}$$

in cui il termine m è la massa, *boldsymbolI* è il tensore di inerzia, mentre i restanti termini sono le accelerazioni provocate dai disturbi.
La distribuzione statistica delle accelerazioni è Gaussiana con le seguenti caratteristiche:

	media μ	dev.std. σ	unità
$\dot{\boldsymbol{V}}_{B,D} = [\dot{u},\, \dot{v},\, \dot{w}]$	0.0	0.005	$\mathrm{m/s^2}$
$\dot{\boldsymbol{\omega}}_D = [\dot{p},\, \dot{q},\, \dot{r}]$	0.0	0.175	$\mathrm{rad/s^2}$

Tabella 5.11. Parametri statistici delle distribuzioni Gaussiane dei disturbi.

Si riporta infine il confronto tra il valore della coppia di disturbo attorno all'asse Y_B elaborata dal modello*Simulink* e in *Python* in figura 5.17, al fine di dimostrare la sovrapponibilità dei disturbi a cui sono sottoposti la *policy* e il regolatore PID.

Il passo fisso di integrazione delle equazioni del moto è di $0.01\,s$ mentre la frequenza operativa del regolatore PID è di $25\,Hz$ come la *policy*. Il campionamento dello stato in retroazione al controllore è effettuato con il metodo *Zero-order Hold*.

5.4 Risultati

La verifica e il confronto delle prestazioni è eseguito in una simulazione di 256 passi temporali corrispondenti a $10.24\,s$.
Il comando è un segnale a gradino imposto sull'angolo di assetto considerato, la risposta per ognuno dei tre angoli è esaminata singolarmente, per cui i risultati corrispondono a tre diverse simulazioni in cui si varia il comando su un angolo lasciando nulli i riferimenti per gli altri due.

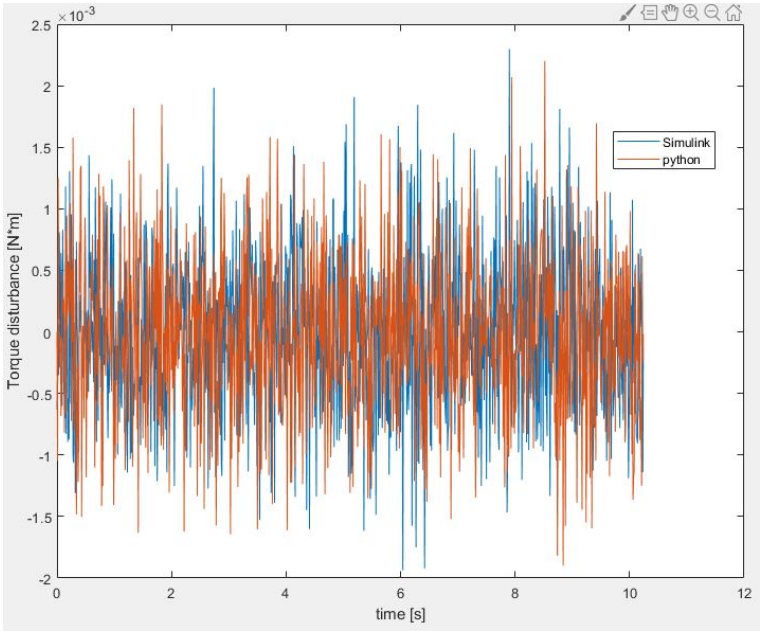

Figura 5.17. Sovrapposizione dei rumori generati dal modello simulato in *Python* e in *Simulink* per il momento di beccheggio.

Nei grafici di figura 5.18 si osserva come le prestazioni a stazionario siano paragonabili per gli angoli ϕ e θ mentre il PID mostra un comportamento molto peggiore a stazionario nel regolare l'angolo di rotta ψ.
In tabella 5.12 sono riportati i valori del *root-mean square error* a stazionario; in generale emerge un miglioramento nell'uso della *policy*, segno di una migliore tolleranza ai disturbi casuali da parte dell'agente rispetto al regolatore lineare. Il miglioramento complessivo per l'errore sugli angoli di assetto è nell'intervallo $45 \div 55\%$.

	RMSE PID	RMSE RL
ϕ	$1.84E-2$	$1.09E-2$
θ	$1.84E-2$	$1.03E-2$
ψ	$1.56E-1$	$6.92E-2$

Tabella 5.12. Valori RMS degli errori sugli angoli di Eulero per regolatore PID e agente.

Il comportamento nel transitorio vede prestazioni nettamente migliori dell'agente rispetto al PID: il tempo di salita è dimezzato e l'*overshoot* è ridotto di un ordine di grandezza sugli angoli ϕ e θ; per l'angolo di rotta invece, le prestazioni dei due tipi di regolazione sono confrontabili anche se il comportamento dell'agente risulta leggermente migliore. Questo risultato è legato anche al fatto che la potenza di controllo sull'asse di imbardata è minore rispetto a quella sugli altri due assi dato che il comando sul momento di imbardata è attuato sfruttando la coppia dei rotori.

	T. salita PID $[s]$	T. salita RL $[s]$	*overshoot* PID $[^o]$	*overshoot* RL $[^o]$
ϕ	0.4	0.24	0.918	0.034
θ	0.4	0.28	0.918	0.027
ψ	1.0	0.6	0.902	0.734

Tabella 5.13. Tempo di salita e *overshoot* sui tre angoli dato dal regolatore PID e dall'agente.

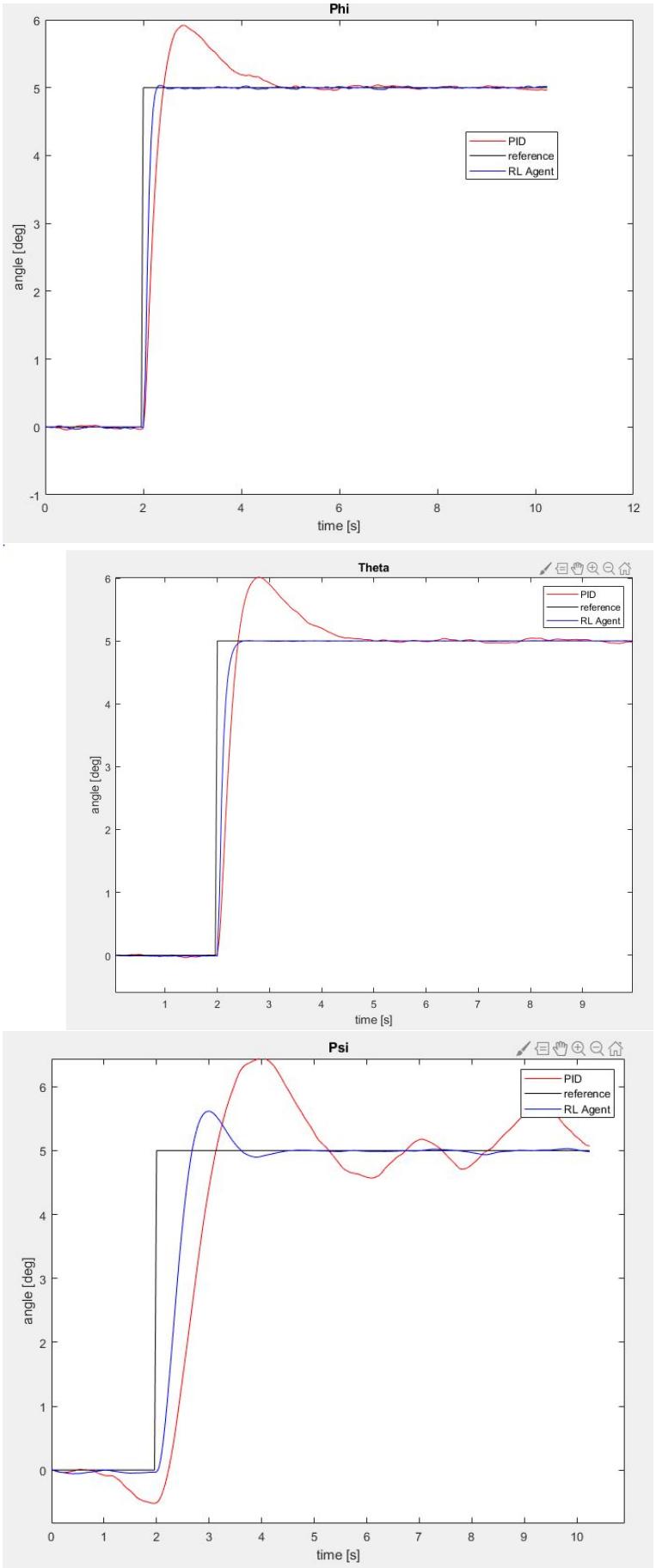

Figura 5.18. Andamento della risposta ad un comando a gradino di 5^o imposto rispettivamente su ϕ, θ e *psi* del sistema controllato con PID e del sistema controllato con RL.

Capitolo 6

Conclusioni

Obiettivo della tesi era di studiare l'applicazione della tecnica del RL per la soluzione del problema del controllo del modello di un quadricottero, attraverso l'analisi delle prestazioni di controllori implementati attraverso questo metodo. L'obiettivo era dimostrare che attraverso questa tecnica si ottengono leggi di controllo in grado di gestire il sistema a partire da condizioni iniziali estreme, capaci di gestire disturbi casuali che rappresentano tubolenze aerodinamiche e in grado di ottenere prestazioni superiori rispetto a quelle di controllori tradizionali.

E' stato implementato il modello di simulazione della dinamica del mezzo e il metodo di integrazione in ambiente *Python*, seguito da alcune considerazioni sull'accuratezza della modellazione degli effetti aerodinamici sui rotori.
E' stato impostato l'allenamento delle reti neurali, interfacciando il modello di simulazione con l'agente per ognuno dei controllori da realizzare. Sono stati ottenuti tre agenti per effettuare tre differenti modalità di controllo: posizionamento del quadricottero a partire da posizione ed assetto casuale, controllo per la navigazione vettoriale e controllo di assetto.
Per ogni controllore è stata selezionata la composizione del vettore delle osservazioni da fornire in input all'agente.
I controllori ottenuti sono stati validati singolarmente, sviluppando per ognuno un test simulato specifico. E' stata realizzata una prova Monte Carlo per verificare le prestazioni del controllore di posizione a partire da diversi stati iniziali; è stata dimostrata la capacità di raggiungere il punto stabilito anche con velocità iniziali nell'ordine dei $4\,m/s$.

E' stata realizzata la simulazione di guida vettoriale impostando una missione consistente nel raggiungimento di 4 *waypoint* con decollo e atterraggio verticale; il controllore ha completato correttamente la missione compensando anche dei disturbi casuali sotto forma di accelerazioni lineari ed angolari con distribuzione Gaussiana.
Per la realizzazione di questa prova, l'agente per la guida vettoriale è stato interfacciato con un regolatore proporzionale che genera i riferimenti vettoriali a partire dall'errore di posizione; è stato cosi realizzato un controllore gerarchico con un anello di basso livello basato su rete neurale e un anello di livello intermedio basato su un regolatore lineare.

E' stata impostata la simulazione di risposta ad un segnale di riferimento a gradino per la validazione del controllore di assetto; è stato riportato il modello di simulazione in ambiente *Simulink* ed è stato realizzato un controllore di assetto di tipo PID per effettuare un confronto di prestazioni. Sia il regolatore PID che l'agente dovevano gestire delle perturbazioni modellate come accelerazioni lineari ed angolari casuali con distribuzione Gaussiana.
In seguito all'osservazione dei risultati del confronto si conclude che l'agente ha prestazioni superiori rispetto al PID sia nella gestione dei disturbi a stazionario sia nel transitorio: è stato osservato un dimezzamento dei tempi di salita, un annullamento dell'*overshoot* in particolare sugli angoli ϕ e θ ed una riduzione del 40% del RMSE a stazionario.

6.0.1 Problematiche delle tecniche di apprendimento automatico

I problemi legati all'uso delle tecniche proposte individuati nel presente lavoro sono principalmente due. In primo luogo si evidenzia la necessità di sviluppare un modello di simulazione del sistema da controllare. Infatti, anche se concettualmente la tecnica consente di determinare i parametri della rete neurale che implementa la funzioni di controllo applicando dal principio quest'ultima sul sistema, tale procedura non è applicabile nella pratica. I parametri della rete sono inizializzati a valori casuali pertanto nelle prime fasi di applicazione dell'algoritmo di RL, la funzione di controllo è totalmente inefficace. Se il processo di apprendimento venisse impostato direttamente su un velivolo reale si incorrerebbe certamente in diversi incidenti prima che l'algoritmo riesca ad aggiornare i parametri della funzione di controllo in modo da ottenere un controllo efficace. Nel presente lavoro dunque è sviluppato il modello di simulazione su cui vengono impiegate le reti neurali al fine di raccogliere i dati per la determinazione dei parametri. La validazione delle prestazioni dei controllori ottenuti è eseguita sempre in un ambiente simulato. Tale procedura comporta che le prestazioni del controllore ottenuto dipendono dall'accuratezza del modello su cui è stato applicato per la definizione dei parametri. Tuttavia un vantaggio rispetto alle tecniche di controllo *model-based* è che il metodo proposto consente di effettuare una definizione dei parametri preliminare sul modello di simulazione e, in seguito, di proseguire l'aggiornamento della legge di controllo in base alle prestazioni ottenute sul velivolo reale.

Il secondo problema legato all'uso di queste tecniche è analizzare la robustezza del controllore ottenuto. Per funzioni di controllo implementate con reti neurali risulta infatti impossibile determinare a priori l'uscita in qualsiasi condizione operativa o effettuare analisi di risposta in frequenza per ottenere margini di fase e di guadagno del sistema. Tale aspetto non rappresenta una limitazione nel momento in cui questi metodi sono sfruttati in un contesto di ricerca su mezzi *unmanned* ma comporta l'impossibilità di estendere i risultati ottenuti al settore dell'industria aeronautica. Data la crescente diffusione della tecnica del RL applicata al controllo di sistemi dinamici, diversi lavori di ricerca sono stati presentati sul tema della robustezza dei controllori ottenuti applicando questi metodi. In particolare si cita il lavoro [41] dove si propone un metodo per determinare la robustezza di una rete neurale conoscendo

soltanto il valore dei suoi parametri, senza la necessità di osservare il comportamento della stessa per tutti i possibili segnali di input; nel lavoro si sottolinea come questo metodo risulta sempre più costoso e inefficiente data della crescente complessità dei sistemi basati sui metodi di apprendimento autonomo.

6.1 Sviluppi futuri

Si evidenziano alcuni spunti per futuri lavori sul tema proposto nella tesi:

- sviluppo di un sistema basato su agenti allenati tramite RL per l'identificazione completa di ostacoli fissi e mobili da integrare in un algoritmo di generazione automatica della traiettoria;
- identificazione del modello di simulazione dei velivoli attraverso l'uso del RL: in questo tipo di applicazione la rete neurale costituisce il modello di simulazione del velivolo e la definizione dei parametri è eseguita allo scopo di ottenere dalla *policy* le funzioni di trasferimento del velivolo reale;
- implementazione di sistemi di gestione del volo (FMS) completi che prevedano la possibilità di ingaggiare diversi controllori basati sul metodo del RL per il completamento di missioni di volo articolate;
- ottimizzazione automatica dei guadagni di controllori lineari attraverso agenti allenati con RL; in questo caso l'agente riceve in ingresso i dati prestazionali della legge di controllo lineare e restituisce in uscita l'aggiornamento dei guadagni del controllore;
- elaborazione di metodi per determinare la robustezza dei controllori basati sulle tecniche di apprendimento autonomo.

Bibliografia

[1] R. Sutton and A. Barto. *Reinforcement Learning: an Introduction. Second edition.* 2018.

[2] Y. Zhou. *Online Reinforcement Learning Control for Aerospace Systems.* PhD thesis, T.U. Delft, 2018.

[3] P. Abbeel, A. Coates, and A.Y. Ng. Autonomous helicopter aerobatics through apprenticeship learning. *The International Journal of Robotics Research*, 2010.

[4] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter. Control of a quadrotor with reinforcement learning. *IEEE Robotics and Automation letters*, 2017.

[5] A. Zavoli and L. Federici. Reinforcement learning for low-thrust trajectory design of interplanetary missions. 2020.

[6] C.H. Pi, K.C. Hu, S. Chegng, and I.C. Wu. Low-level autonomous control and tracking of quadrotor using reinforcement learning. *Control Engineering Practice*, 2020.

[7] G. Apostolo. *The Illustrated Encyclopedia Of Helicopters.* Bonanza Books, 1984.

[8] P. Lambermont. *Helicopters and Autogyros of The World.* Cassell, 1956.

[9] S. Harding. *U.S. Army Aircraft Since 1947.* Specialty Pr Pub & Wholesalers, 1990.

[10] J. Kim, A. Gadsden, and S. Wilkerson. A comprehensive survey of control strategies for autonomous quadrotors. *IEEE*, 2020.

[11] A.R.S. Bramwell, G. Done, and D. Balmford. *Bramwell's Helicopter Dynamics.* Butterworth-Heinemann, 2001.

[12] P. Marqués and A. Da Ronch. *Advanced UAV Aerodynamics, Flight Stability and Cntrol.* Wiley, 2017.

[13] https://www.unmannedsystemstechnology.com/royal-navy-tests-remotely-piloted-systems-in-man-overboard-trials.

[14] P. Ghosh, A. Gasparri, J. Jin, and B. Krishnamachari. Robotic wireless sensor networks. *arXiv*, 2017.

[15] EASA. Regulation 2019/945 and 2019/947: Rules for unmanned aircraft systems, 2019. https://www.easa.europa.eu/document-library/easy-access-rules/easy-access-rules-unmanned-aircraft-systems-regulation-eu.

[16] D. Arena. Sviluppo del sistema di guida e controllo di un quadrotor. Master's thesis, Università La Sapienza, A.A.2015/2016.

[17] E.M. Napoleoni. Sviluppo del sistema di guida e controllo di un quadrotor. Master's thesis, Università La Sapienza, A.A.2014/2015.

[18] S. Bouabdallah, A. Noth, and R. Siegwart. Pid vs lq control techniques applied to an indoor micro quadrotor. In *In Proceedings of 2004 1EEElRS.J Internationel Conference On Intelligent Robots and Systems.*

[19] F. Kendoul. Survey of advances in guidance, navigation, and control of unmanned rotorcraft systems. *Journal of Field Robotics*, 2012.

[20] P. Bolzern, R. Scattolini, and N. Schiavoni. *Fondamenti di Controlli Automatici.* McGraw-Hill, 2004.

[21] D.J. Leith and W.E. Leithead. *Survey of Gain-Scheduling Analysis and Design.* PhD thesis, University of Strathclyde, 2012.

[22] H. Mo and G. Farid. Non linear and adaptive intelligent control techniques for quadrotor uav - a survey. *Asian Journal of Control*, 2019.

[23] S. Waslander, G. Hoffmann, J.S. Jang, and C. Tomlin. Multi-agent quadrotor testbed control design: Integral sliding mode vs. reinforcement learning. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems.*

[24] N. Bernini, M. Bessa, R. Delmas, A. Gold, E. Goubault, R. Pennec, S. Putot, and F. Sillion. A few lessons learned in reinforcement learning for quadcopter attitude control. In *24th ACM International Conference on Hybrid Systems: Computation and Control (HSCC '21).*

[25] R. Sutton and A. Barto. *Reinforcement Learning: an Introduction. Second edition*, chapter 1: Introduction.

[26] R. Sutton and A. Barto. *Reinforcement Learning: an Introduction. Second edition*, chapter 6: Temporal Difference Learning.

[27] J. Schulmann, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *OpenAI*, 2017.

[28] C. Zhang, O. Vinyals, R. Munos, and S. Bengio. A study on overfitting in deep reinforcement learning. *arXiv*, 2018.

[29] R. Sutton and A. Barto. *Reinforcement Learning: an Introduction. Second edition*, chapter 13: Policy Graient Methods.

[30] https://spinningup.openai.com/en/latest/spinningup/rl_ intro3.html.

[31] https://stable-baselines.readthedocs.io/en/master/.

[32] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007.

[33] B. Etkin and L.D. Reid. *Dynamics of Flight, Stability and Control*. Wiley, 1996.

[34] W.L. Hankey, L.E. Miller, and S.J. Scherr. Use of quaternions in flight mechanics. Technical report, Air Force Wright Aeronautical Laboratories Wright-Patterson Air Force Base, OH, 1984.

[35] A. Sulejmani. Generazione e inseguimento di traiettoria per sciami di quadrirotori. Master's thesis, Università La Sapienza, A.A.2019/2020.

[36] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a quad-rotor robot. In *Proceedings of the 2006 Australasian Conference on Robotics and Automation*.

[37] H. Bouadi and M. Tadjine. Nonlinear observer design and sliding mode control of four rotor helicopter. In *Proceedings of World Academy of Science, Engineering And Technolody Volume 25 November 2007*.

[38] T. Lukkonen. Modelling and control of quadcopter.

[39] E. Fresk and G. Nikolakopoulos. Full quaternion based attitude control for a quadrotor. *European Control Conference*, 2013.

[40] N. Michael, D. Mellinger, Q. Lindsay, and V. Kumar. The grasp multiple micro-uav test bed experimental evaluation of multirobot aerial control algorithms. *IEEE Robotics and Automation Magazine*, 2010.

[41] E. El Mhamdi, R. Guerraoui, and S. Rouault. On the robustness of a neural network. In *36th IEEE International Symposium on Reliable Distributed Systems 26 - 29 September 2017. Hong Kong, China*.