Alma Mater Studiorum Università di Bologna

SCUOLA DI INGEGNERIA
Dipartimento di Ingegneria Industriale
Corso di Laurea in Ingegneria Aerospaziale

Analisi e sperimentazioni per l'utilizzo ed il controllo di un microdrone in ambiente indoor

Relatore:
Prof. Matteo Zanzi

Presentata da:
Lorenzo Balzani

Correlatore:
Dott. Massimiliano Menghini

Anno Accademico 2022/2023

A tutta la mia famiglia

Sommario

Gli aeromobili a pilotaggio remoto, comunemente noti come "droni" hanno fatto molta strada dalla loro prima ideazione e, specialmente negli ultimi decenni, sono andati incontro a uno sviluppo a ritmi sempre più elevati, passando da meri strumenti militari a dispositivi utilizzati per l'esplorazione, lo sport, la fotografia, la consegna di pacchi e molto altro.

Grazie alla crescente facilità di accesso alle nuove tecnologie, inoltre, è sempre più comune la realizzazione di piccoli e relativamente economici apparecchi, utilizzabili anche in ambiente domestico e da persone sprovviste di abilitazioni specifiche del settore.

Ne è un esempio il "Crazyflie 2.1" prodotto dall'azienda svedese "Bitcraze", su cui si basa questa ricerca. Quest'ultimo dispositivo è caratterizzato dalle sue dimensioni contenute e dalla possibilità, essendo open source, di effettuare sviluppi e modifiche ai software successivamente all'acquisto. In aggiunta al drone stesso, sono disponibili per l'utilizzo altri sistemi e pacchetti di espansione offerti da Bitcraze per l'ampliamento delle funzionalità di Crazyflie 2.1, fra questi troviamo il "Flow Deck v2", il "Z-ranger deck v2" e il "Multi-ranger Deck", di cui verranno approfondite le caratteristiche in seguito.

La presente trattazione mira a fornire informazioni ed esempi riguardanti le caratteristiche e possibilità applicative di questo particolare dispositivo.

Contenuti

Introduzione

Secondo l'enciclopedia "Treccani" un drone può essere definito come "un velivolo privo di pilota e comandato a distanza" [1], sebbene questa spiegazione sia accurata, è ampio lo spazio lasciato all'interpretazione, per questo motivo, nel corso della Storia, i cosiddetti "droni" sono andati incontro ad una evoluzione e contestuale cambiamento degno di nota.

Un primo tentativo rilevante di utilizzo di dispositivi aerei privi di equipaggio per fini bellici è risalente al 1849, quando, gli austriaci, intenti a piegare la resistenza della Repubblica di San Marco, utilizzarono delle mongolfiere cariche di esplosivo collegato a micce a tempo per effettuare quello che verrà ricordato come "il primo tentativo di bombardamento aereo della storia" sulla città di Venezia.

L'uso di palloni nelle compagnie militari non era del tutto nuovo, ma in passato erano stati utilizzati solo per scopi ricognitivi e scambio di informazioni.

Seppur l'operazione si rivelò un fallimento, in quanto il posizionamento corretto dei palloni che avrebbero dovuto sganciare le bombe incendiarie venne affidato al vento, si può pensare a questo avvenimento come un punto di inizio per una tendenza dal punto di vista degli armamenti che caratterizzò gli anni seguenti.

Fu la prima guerra mondiale a dare la scintilla per lo sviluppo di queste nuove e innovative tecnologie. Nel corso del 1917, infatti, vennero ideati e testati dalla Royal Flying Corps, ovvero la forza aerea del Regno Unito in carica, i primi velivoli senza pilota comandati tramite onde radio, conosciuti con il nome di "Aerial Target". L'anno seguente anche il "Kettering Bug", un missile sperimentale delle forze armate americane, effettuò con successo i primi test di volo.

Benché promettenti, nessuno di questi due sistemi trovò mai applicazione durante la guerra, nonostante questo si continuò a sviluppare e testare nuove tecnologie per la creazione di rinnovati e maggiormente efficaci aeromobili senza pilota.

Nel corso degli anni trenta vennero prodotti e utilizzati i cosiddetti "Target drone" ovvero dei droni bersaglio. Molto simili a un normale aeromobile dell'epoca, ma sprovvisti di equipaggio, i target drone venivano controllati a distanza tramite onde radio, e svolgevano la funzione di obiettivo per le esercitazioni militari delle forze armate.

Al riguardo, vale la pena citare il "Curtiss N2C", già in dotazione alla marina militare statunitense, venne modificato e testato nel 1937 per permettere il controllo da remoto. Di maggiore notorietà è il "OQ-2 Radioplane" un target drone di dimensioni ridotte che venne prodotto in diverse migliaia di esemplari negli Stati Uniti durante i primi anni della Seconda Guerra Mondiale.

Le cose si mossero anche sul fronte britannico che nel 1935 testò il drone bersaglio "Queen Bee", anche in questo caso si trattava di un aeromobile preesistente modificato per permettere il controllo da distanza.

Nel corso del Secondo Conflitto Mondiale vennero inoltre utilizzati, seppur in maniera sporadica, quelli che possiamo considerare come i primi droni da assalto della storia. Lo statunitense "TDR-1", difatti, era in grado di essere armato e sganciare bombe o siluri una volta in volo.

Di maggior impiego fu il "Fieseler Fi 103" considerato come il primo missile da crociera della storia. Prodotto in diverse decine di migliaia di esemplari ed utilizzato dalle Forze aeree tedesche, questo sistema era in grado di essere teleguidato e di percorrere fino a 240 chilometri.

Fu durante la Guerra in Vietnam, tuttavia, che i droni vennero consacrati come elemento essenziale ed imprescindibile di un esercito all'avanguardia. Le forze aeree americane, infatti, preoccupate di perdere piloti in territorio nemico, iniziarono ad utilizzare in maniera massiccia aeromobili sprovvisti di equipaggio fra cui possiamo citare il "Ryan Model 147" e il " Lockheed D-21", dei droni con motori a getto utilizzati per la ricognizione in terra ostile. Alla conclusione del conflitto nel 1975 più di 3400 missioni erano state svolte da apparecchi controllati da remoto.

Nei decenni successivi, grazie allo sviluppo e alla miniaturizzazione dei sistemi utilizzabili, i droni hanno visto un'evoluzione ed utilizzo considerevole, fino ad arrivare a congegni con capacità offensive, di controllo e di autonomia ragguardevoli. Degli esempi di dispositivi di ultima generazione tristemente noti per il conflitto Russo-Ucraino sono il "Bayraktar TB2", sfruttato per la sorveglianza e lo svolgimento di azioni militari utilizzando le bombe e i missili con cui è equipaggiato, ed il "Geran-2", un drone suicida adoperato per colpire bersagli a terra con grande precisione.

Fortunatamente il progresso relativo alle tecnologie e ai processi industriali ha permesso che i droni diventassero gradualmente sempre più piccoli, leggeri ed economici, rendendoli così appetibili per una varietà di applicazioni al di fuori delle organizzazioni militari.

Fra esse possiamo citare:

- Cinema e Fotografia: Grazie alle loro capacità di ripresa aeree, infatti, i droni stanno diventando sempre più utilizzati da registri e fotografi.

- Agricoltura: In questo caso i droni hanno il ruolo di rilevazione e monitoraggio dei raccolti ed eventualmente sono dotati anche di sistemi di spargimento per contrastare i parassiti.

- Spettacolo: Sono molti gli esempi nella storia recente di utilizzo di sciami di droni per creare spettacoli unici grazie alle luci a LED di cui sono dotati.

- Consegne: Sebbene sia ancora un settore in via di sviluppo, per le molteplici problematiche, fra cui quelle relative al controllo del traffico aereo, è possibile pensare che in futuro i droni saranno utilizzati per svolgere consegne rapide ed efficienti ai clienti delle grandi aziende di commercio.

- Operazioni di soccorso: Grazie alle loro capacità di raggiungere zone ostiche per l'uomo, i droni sono molto utilizzati per la localizzazione di persone in pericolo o il trasporto di risorse. Ad oggi si stima che le persone salvate grazie all'utilizzo di questi dispositivi siano più di 1000 [2].

Contestualmente al crescente interesse per i droni scaturito nel corso del XXI secolo, sempre più dispositivi utilizzati in ambito sportivo o ricreativo sono stati realizzati. Solo negli Stati Uniti i droni registrati risultano essere più di 860000, di cui solo circa 350000 sono utilizzati a scopi commerciali [3].

L'ampliamento e l'evoluzione di questo mercato ha portato, inoltre, allo sviluppo anche di piccoli apparecchi utilizzabili in ambienti chiusi, ed è proprio su uno di questi che si basa questa ricerca.

L'obiettivo della presente trattazione è quello di studiare ed effettuare sperimentazioni sulle possibilità ed il comportamento del drone "Crazyflie 2.1".

L'elaborato è stato strutturato in 3 capitoli.

Nella prima sezione sono raccolte tutte le informazioni riguardanti il drone, i sensori e l'equipaggiamento montato.

Il secondo segmento è invece dedicato alle prove sperimentali riguardanti l'invio di istruzioni al drone tramite codici in python. In questo capitolo verranno perciò presentati degli script rappresentanti delle possibilità applicative per l'utilizzo del quadricottero e dei suoi sensori.

Nel terzo paragrafo verranno infine fatte alcune brevi considerazioni sulle caratteristiche di stabilità di Crazyflie 2.1.

Capitolo 1

Informazioni preliminari

1.1 Equipaggiamento utilizzato

- Il drone Crazyflie 2.1

- Crazyradio 2.0 : Una chiavetta USB radio a lunga distanza [4], usata per trasmettere e ricevere informazioni dal drone alla stazione a terra e viceversa.

- Bitcraze Virtual Machine: Una macchina virtuale creata dall'azienda produttrice del drone (Bitcraze) e contenente diverse applicazioni e file utili per l'utilizzo, il pilotaggio e lo sviluppo di Crazyflie 2.1. All'interno della macchina, fra le altre, sono disponibili un Client, che permette di visualizzare i dati ricevuti dal drone ed inviare comandi, e tutti i codici dei progetti di Bitcraze, fra cui il firmware del drone.

- Expansion decks (Flow deck v2, Multi-ranger deck, Z-ranger deck v2) : Sono piattaforme hardware che possono essere attaccate al drone per ampliarne le funzionalità e i sensori a disposizione.

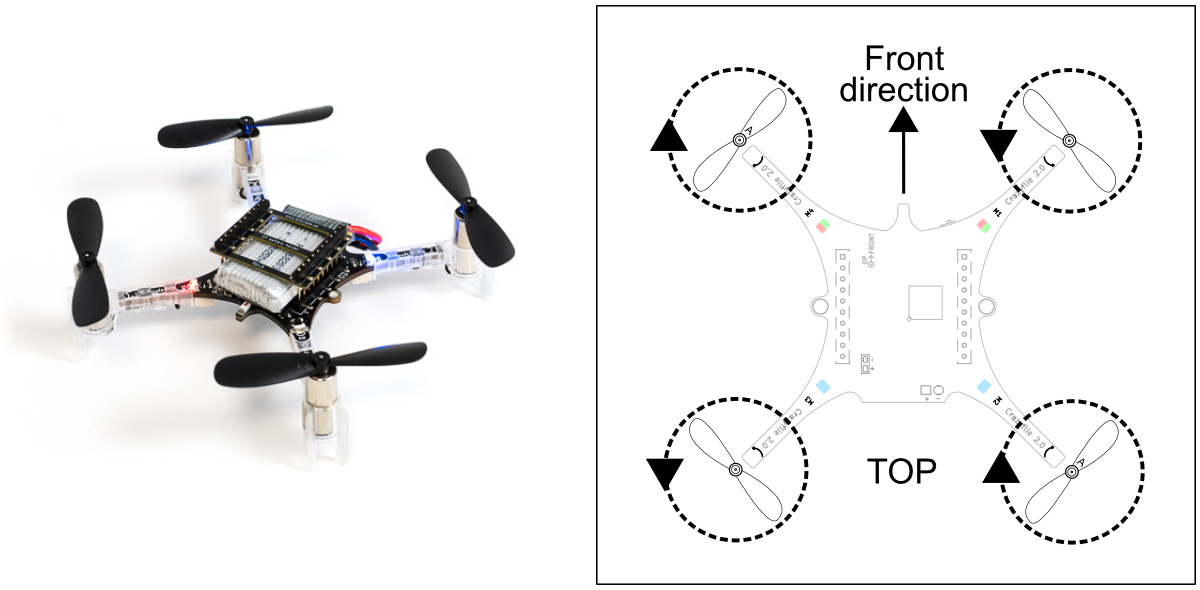
1.2 Caratteristiche del drone

"Crazyflie 2.1" (fig. 1.1a) è una versatile piattaforma volante open source [5], prodotta dall'azienda svedese Bitcraze. Il drone ha dimensioni molto contenute (92mm x 92mm x 29mm) e un peso al decollo di 27g, per questo motivo si presta principalmente ad un utilizzo in ambiente indoor.

Il dispositivo è dotato, inoltre, di un sistema radio a bassa latenza e ampia distanza (circa 1 chilometro) oltre che di uno Bluetooth che permette il collegamento per il pilotaggio con la relativa applicazione su cellulare.

I propulsori sono costituiti da eliche di 45mm di diametro, trascinate da motori coreless a corrente continua, dotati di connettore alle estremità e capaci di elargire 14000 giri al minuto/volt con un voltaggio nominale di 4.2 Volt e una corrente nominale di 1000mA.

Il drone presenta una configurazione di tipo Quadricottero a X con versi di rotazione delle eliche speculari come in figura 1.1b, in modo da poter garantire un bilanciamento attorno all'asse di imbardata.



(a) (b)

Figura 1.1: Crazyflie 2.1

Per finire come accumulatore di energia elettrica viene utilizzata una batteria litio-polimero (LiPo) con capacità di 250mAh e massa di 7.1g, la quale garantisce una autonomia di volo di circa 7 minuti.

1.3 Sensoristica drone

Così come moltissimi altri velivoli specialmente nel settore aeronautico, anche Crazyflie 2.1 è dotato di un IMU (Inertial Measurment Unit), ovvero un'unità di misura inerziale, composta da una terna di accelerometri e giroscopi ortogonali (più informazioni a riguardo in appendice A) e da un sensore di pressione ad alta precisione.

1.3.1 IMU

Per quanto riguarda l'unità di misura inerziale, il drone è equipaggiato con il "BMI088" dell'azienda Bosch (fig. 1.2a).

BMI088 è in grado di rilevare movimenti e rotazioni nei sei gradi di libertà, esso combina le funzionalità di un giroscopio triassiale da 16bit e di un accelerometro, anch'esso triassiale e da 16bit, in un unico dispositivo [6].

Entrambi i sensori inerziali sono di tipo MEMS (Micro electro-mechanical systems), grazie anche a questa caratteristica il pacchetto è di dimensioni molto contenute (3mm x 4.5mm x 0.9mm) ed adatto all'utilizzo su droni e robot, per merito della sua alta resistenza alle vibrazioni. In virtù della possibilità di misurare accelerazioni fino a 24g, infine, il dispositivo garantisce range di utilizzabilità molto ampi.

1.3.2 Sensore di Pressione

Crazyflie 2.1 è dotato di "BMP388" anch'esso realizzato dall'azienda "Bosch", un sensore digitale in grado di fornire una misura della pressione e della temperatura [7].

Il dispositivo sebbene di dimensioni contenute (2.0mm x 2.0mm x 0.75mm) e con un basso consumo energetico, garantisce un ampio range di funzionamento (da 300 a 1250 hPa).

In particolare BMP388 consiste in un sensore di pressione piezoresistivo (più informazioni a riguardo in appendice B) e un ASIC (Application Specific Integrated Circuit).

1.4 Microcontrollori a bordo

Il drone è equipaggiato con 2 microcontrollori, anche noti come MCU (MicroController Unit): "STM32F405" e "nRF51822".

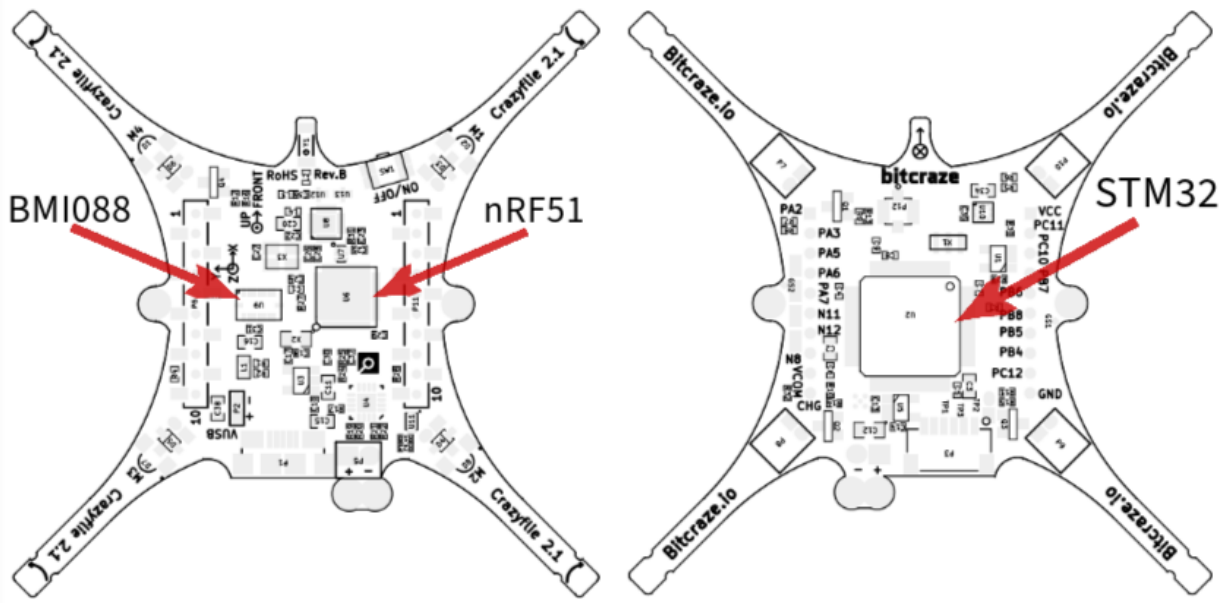
"STM32F405" (fig. 1.2b) è l'MCU a cui sono affidate le applicazioni principali, fra cui il pilotaggio. Si tratta di un dispositivo progettato dall'azienda italo-francese "STMicroelectronics NV", che offre alte prestazioni, sistemi di memoria embedded e un ricco set di periferiche, in un pacchetto di dimensioni molto ridotte [8].

In particolar modo l'apparato è dotato del core (ovvero il nucleo elaborativo del microprocessore) "Arm® Cortex®-M4" a 32-bit e di tipo RISC (Reduced Instruction Set Computer), capace di operare a frequenze fino a 168 MHz. Il core è inoltre dotato di una unità di calcolo in virgola mobile (FPU). Nel microcontrollore sono infine incorporate anche memorie embedded ad alta velocità che garantiscono fino a 1 Mbyte di memoria flash e fino a 192 kbytes di SRAM (Static Random Access Memory).

"nRF51822" (fig. 1.2a) è l'MCU usato principalmente per l'apparato radio, le comunicazioni e la gestione e distribuzione della potenza. Esso è un sistema su circuito integrato (SoC) wireless a bassa potenza a cui è integrato un ricetrasmettitore radio da 2.4 GHz, una CPU (Central Processing Unit) "ARM® Cortex™ M0", 128 kbytes di memoria flash e 16kbytes di SRAM, oltre che periferiche sia analogiche che digitali [9].

Il dispositivo è inoltre in grado di supportare "Bluetooth Low Energy", una tecnologia wireless utile per il collegamento con cellulari.

Il drone è infine dotato di un EEPROM (Electrically Erasable Programmable Read-Only Memory) da 8Kbytes. Questo dispositivo di memoria è utile per registrare dati che devono essere mantenuti anche una volta scollegata l'alimentazione elettrica.



(a) vista dall'alto (b) vista dal basso

Figura 1.2: piattaforma Crazyflie 2.1

1.5 Deck di espansione

Crazyflie 2.1 è stato progettato per essere un dispositivo molto flessibile, con ampie possibilità di espansione e sviluppo, effettuabili anche autonomamente dal possesore del drone.

Per questo motivo l'apparecchio è dotato di due interfacce (nella parte superiore ed inferiore del drone) in cui è possibile attaccare dei deck di espansione, ovvero delle piattaforme che forniscono a Crazyflie 2.1 delle funzionalità aggiuntive, come un sistema di posizionamento assoluto o relativo, sensori addizionali, maggiore capacità di calcolo computazionale e molto altro.

Durante lo svolgimento di questa ricerca erano disponibili per l'utilizzo i seguenti expansion decks:

- Flow Deck v2
- Multi-Ranger Deck
- Z-ranger Deck v2

di cui verrano riportate le specifiche nelle pagine seguenti. Ai fini della sperimentazione per questo elaborato, tuttavia, sono stati utilizzati principalmente il "flow deck v2" e il "multi-ranger deck".

1.5.1 FLow deck v2

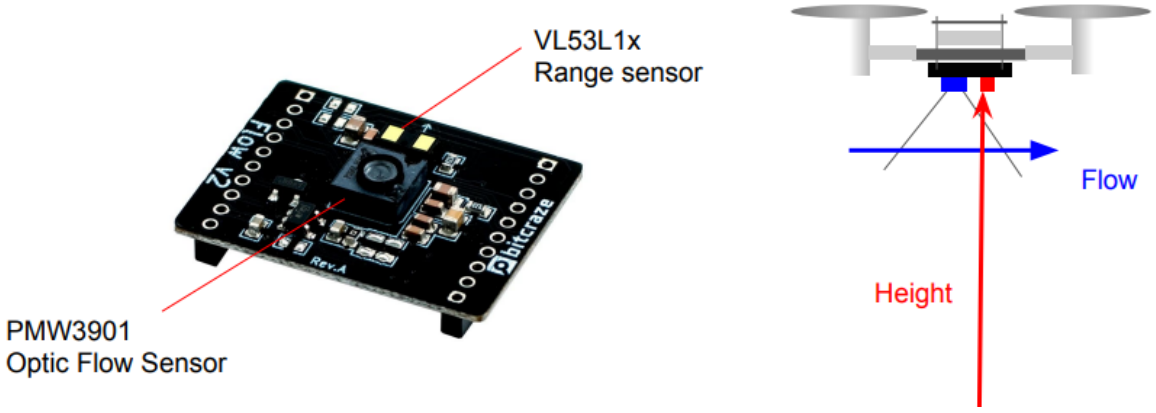


Figura 1.3: Flow deck v2 [10]

Il Flow deck v2 (fig. 1.3) è un dispositivo di 1.6g e 21mm x 28mm x 4mm che fornisce al drone l'abilità di percepire quando e quanto si sta muovendo in una direzione [11].

Il dispositivo è dotato di due sensori: "VL53L1x ToF" che misura la distanza con il terreno e "PMW3901" che misura l'ampiezza dei movimenti in relazione con il suolo.

In particolar modo "VL53L1x ToF" è un sensore laser Tof (Time of Flight) prodotto da "STMicroelectronics NV", capace di misurare distanze con buona precisione fino a 4m e con alte frequenze (fino a 50Hz) [12]. Il funzionamento del rilevatore, in questo caso, si basa sull'emissione di un raggio laser e della misura del tempo impiegato a ricevere indietro la sua componente riflessa, in questo modo è possibile risalire alla distanza da terra.

Il "PMW3901MB-TXQT" è uno degli ultimi chip di navigazione ottica prodotti dall'azienda "PixArt Imaging Inc.". Il pacchetto, in questo frangente, fornisce informazioni sul movimento lungo x e y, con un range di funzionamento molto ampio (da 80mm fino, potenzialmente, all'infinito) [13].

Il flow deck v2 è, infine, particolarmente utilizzato, in quanto, grazie alle sue caratteristiche, migliora considerevolmente la stabilità durante il volo e fornisce la possibilità di preprogrammare un percorso di volo da far seguire al drone in maniera autonoma.

1.5.2 Multi-ranger deck

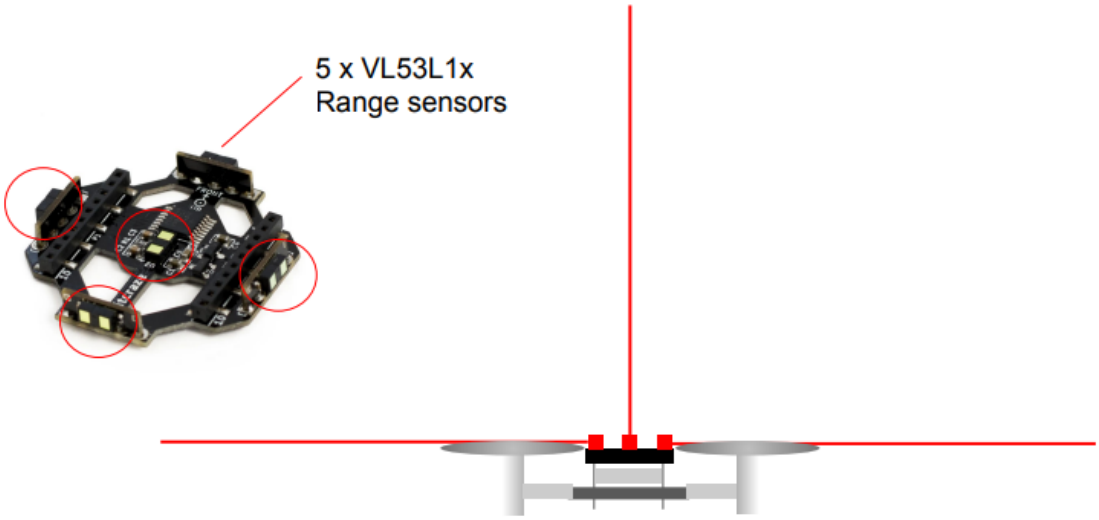


Figura 1.4: Flow deck v2 [10]

Il Multi-ranger deck (fig. 1.4) è una piattaforma di 2.3g e 35mm x 35mm x 5mm che fornisce al drone l'abilità di rilevare gli oggetti circostanti [14]. Per fare ciò il dispositivo utilizza cinque sensori laser "VL53L1x ToF" dell'azienda "STMicroelectronics NV", rivolti in cinque direzioni diverse: verso l'alto, destra, sinistra, avanti e indietro. In questo modo il drone è in grado di individuare ostruzioni fino a 4 metri di distanza nelle direzioni citate con buona precisione (fino a qualche mm, dipendentemente dalle caratteristiche della superficie e dalle condizioni di luce).

"Flow deck v2" e "Multi-ranger deck" sono compatibili per l'utilizzo in contemporanea, in quanto devono essere posizionati in alloggiamenti opposti (Flow deck nella parte inferiore del drone e Multi-ranger in quella superiore).

1.5.3 Z-ranger deck v2

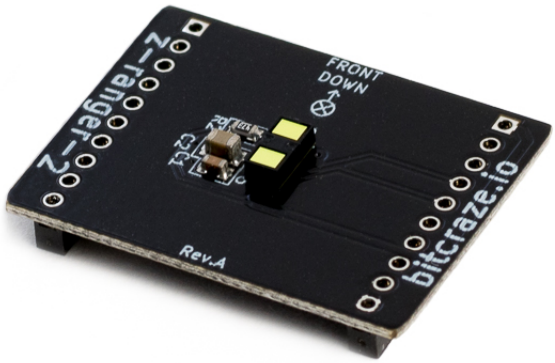


Figura 1.5: Z-ranger Deck v2

Lo Z-ranger deck v2 (fig.1.5) è una piattaforma di 1.6g e 21mm x 28mm x 4mm, essa viene montata nella parte inferiore del Crazyflie 2.1 e fornisce ad esso la possibilità di misurare con precisione, tramite il sensore laser "VL53L1x ToF", la distanza con il terreno [15].

Il deck permette inoltre di mantenere automaticamente un distacco costante dagli oggetti al di sotto del drone durante un volo automatico.

Capitolo 2

Sperimentazioni sulle capacità di controllo autonomo

Per mantenere un collegamento stabile tra drone e stazione a terra è stata utilizzata la Crazyradio 2.0.

Quest'ultima non è un dispositivo utilizzabile solo in concomitanza con altre apparecchiature appartenenti alla famiglia "Crazyflie", bensì è un sistema open project e open firmware che può quindi essere sviluppato ed utilizzato per una varietà di applicazioni.

Il dispositivo è dotato di un python API (Application Programming Interface), per questo è possibile inviare codici python per impartire movimenti o azioni al drone, senza bisogno di essere dotati di un radiocomando.

In questo capitolo verranno esplorate alcune delle possibilità applicative di questa funzione, dotandosi anche di esempi e funzioni, fornite come free software dall'azienda produttrice del drone "Bitcraze".

2.1 Test preparativi

Per poter effettuare questi test è stato utilizzato l'ambiente di sviluppo e apprendimento integrato di Python, ovvero "IDLE3".

Per garantire la sperimentazione in sicurezza, inoltre, sono state svolte delle prove preliminari sulla connessione con il drone e il corretto funzionamento dei deck di espansione montati.

2.1.1 Verifica connessione con drone

Il primo codice utilizzato [16] (Allegato 3.4), garantiva una verifica della connessione con il drone tramite la Crazyradio 2.0.

Nella prima parte dello script vengono importate delle librerie contenenti funzioni necessarie per le operazioni successive. Di queste, "logging" e "time" sono librerie standard di python, mentre "cflib" è la libreria di "Crazyflie" creata dal team di Bitcraze.

La riga successiva definisce l'URI (Uniform Resource Identifier) del drone a cui connettersi. In questo caso la prima porzione dell'identificativo è relativa a impostazioni sulla comunicazione radio.

Nella sezione seguente viene invece definita la funzione che verrà usata nella parte principale del programma (Main). Prima di richiamare questa funzione vengono però inizializzati i drivers usati per le comunicazioni con Crazyflie 2.1 e creata una connessione con esso. La funzione, infine, in mancanza di errori, stampa nel terminale "Connessione stabilita" e, dopo 3 secondi, riporta "Disconnessione" e lo script viene terminato.

2.1.2 Verifica rilevamento Deck di espansione

Nel codice successivo [17] (Allegato 3.5) si verifica che il drone rilevi correttamente il "Flow deck v2", in quanto, per poter volare adeguatamente e garantire una certa stabilità è necessario che sia montato un deck con un sistema di posizionamento (assoluto o relativo).

In questo caso viene monitorato il valore del parametro "bcFlow2" per capire se il deck di espansione sia attaccato opportunamente o no. Nel caso il valore del parametro sia zero nel terminale verrà stampato "Deck non rilevato", in quanto corrispondente a una logica falsa. Viceversa nel caso il valore di "bcFlow2" sia diverso da zero significherà che il deck viene rilevato correttamente pertanto verrà riportato nel terminale il messaggio "Deck rilevato".

2.2 Decollo autonomo

Dopo aver effettuato i test preliminari è stata testata la capacità del drone di effettuare un decollo in autonomia e mantenere in hovering una determinata quota per un periodo di tempo.

Per questo e le successive applicazione è stato utilizzato il "Motion Commander", ovvero una classe in Python creata dal team di Bitcraze per facilitare l'invio di comandi di movimento al drone Crazyflie.

In questo codice (Allegato 3.6), dopo avere importato le opportune librerie e aver stabilito la connessione con il drone, come nei codici precedenti, viene creata un'istanza del "Motion commander" chiamata "mc". Una volta creata quest'ultima il drone decollerà automaticamente, raggiungendo un'altezza di default di 0.3m.

Successivamente tramite il comando "time.sleep(3)" (funzione importata dalla libreria standard di python "time") viene sospesa l'esecuzione del programma per 3 secondi, pertanto il drone resterà in hovering alla quota impostata.

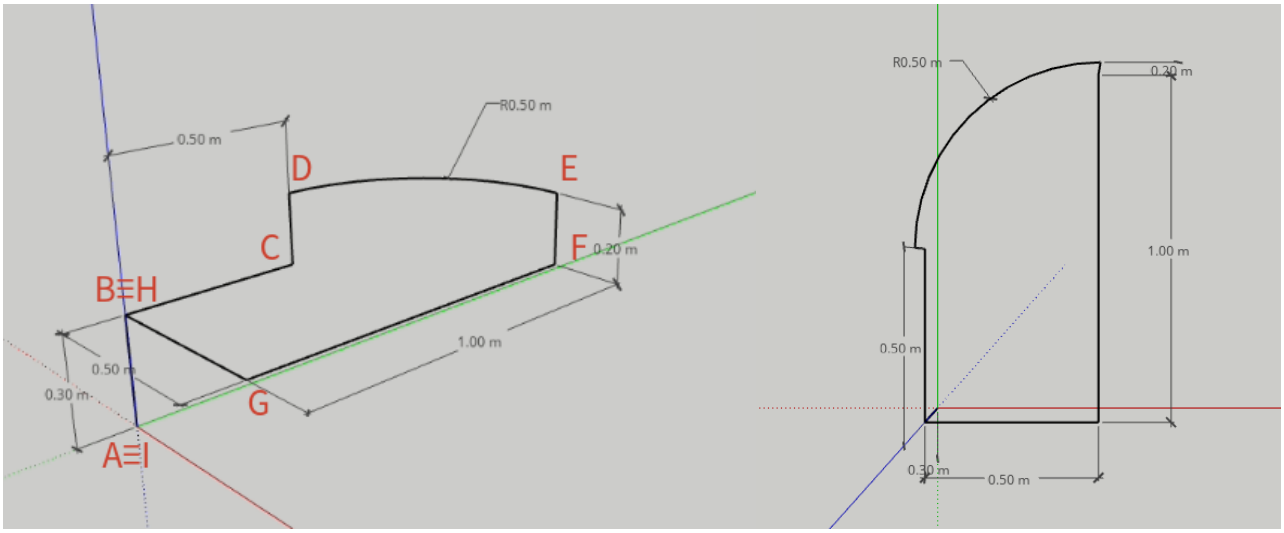
Infine una volta terminato il codice ed usciti dal contesto del Motion commander creato precedentemente con la parola chiave "with", il dispositivo atterrerà automaticamente.

2.3 Percorso di volo preprogrammato

All'interno del Motion Commander creato da Bitcraze sono implementate diverse funzioni per controllare il movimento del drone attorno e lungo i suoi assi. Per questo motivo, a

questo punto della trattazione, si è testata la possibilità di programmazione di un percorso di volo da far seguire al drone.

Il codice implementato in Allegato 3.7, riporta un esempio di percorso preprogrammato come in figura 2.1.



(a) vista 3D (b) vista dall'alto

Figura 2.1: Percorso di volo

Il motion commander utilizza setpoints di velocità e non è dotato di un sistema di posizionamento assoluto, di conseguenza gli errori di posizionamento si accumulano durante il volo. Per questo motivo nel caso di percorsi di volo particolarmente distesi gli errori accumulati possono portare a scostamenti significativi tra la posizione stimata e quella reale.

Per verificare quanto citato si è dunque implementato un codice che fornisse, durante il percorso, le stime dei valori delle coordinate di x e y rilevate dai sensori a bordo del Crazyflie 2.1.

2.4 Stima coordinate x e y percorso preprogrammato

Lo script in questione (Allegato 3.8) utilizza una variabile chiamata "position_estimate", inizializzata all'origine (0,0), per registrare i valori delle coordinate fornite ogni secondo dal quadricottero.

Il percorso preprogrammato è lo stesso analizzato nella sezione precedente (2.3), in questo caso viene definita una funzione "percorso_volo" che contiene le istruzioni per effettuare il tragitto.

Nella parte finale del codice viene quindi avviata la registrazione della stima delle coordinate e dato il comando per avviare il percorso di volo.

Di seguito, in tabella 2.1, si riportano i dati ricevuti corrispondenti ai punti contrassegnati in figura 2.1.

Stima coordinate x y		
Checkpoints percorso	State Estimate X [m]	State Estimate Y [m]
Punto A	0.0017	-0.0011
Punto B	0.0060	-0.030
Punto C	0.48	-0.030
Punto D	0.52	-0.012
Punto E	0.94	-0.48
Punto F	0.90	-0.54
Punto F*	0.88	-0.46
Punto G	-0.12	-0.37
Punto H	-0.14	0.094
Punto I	-0.13	0.10

Tabella 2.1: Valori stimati dal drone delle coordinate x e y

(F* corrispondente alla stima delle coordinate, nel punto F, a seguito del comando di rotazione di 90 gradi attorno all'asse di imbardata del drone)

Com'è possibile notare dai dati forniti ogni movimento presenta un certo scostamento rispetto alla distanza richiesta dal comando inviato. Fra le funzioni che hanno causato maggiori differenze è possibile citare la rotazione attorno all'asse di imbardata (passaggio da checkpoint F a F*) e l'arco di circonferenza. È inoltre possibile notare come le coordinate di decollo (punto A) e le coordinate di atterraggio (punto I) non coincidano per via degli errori accumulati.

Vale la pena sottolineare, infine, che, in quanto stime, anche i valori riportati risultano affetti da errori, i quali, sommati, hanno portato a una discrepanza tra le coordinate fornite dal drone e quelle reali in cui era posizionato.

Al punto di atterraggio (punto I), di fatti, il quadricottero risultava essere approssimativamente a coordinate reali

$$(X = -0.24, Y = 0.23)$$

2.5 Demo "ranging bundle"

Fra gli ultimi test è stato sperimentato un codice implementato dal team di Bitcraze [18] che utilizza in combinazione il Multi-ranger deck, per il rilevamento dell'avvicinamento di possibili ostacoli al drone, e il Flow deck, per consentire un volo autonomo e hovering stabile.

Utilizzando questo script è possibile far in modo che il drone, dopo aver percepito la presenza di ostacoli entro 0.2m in una delle 5 direzioni coperte dal Multi-ranger deck, avanzi nella direzione opposta in modo da evitare la collisione.

Il codice che permette l'utilizzo di questa funzionalità (Allegato 3.9) è strutturato in diverse parti.

Nella prima parte oltre che alcune delle librerie standard di python vengono importati i moduli per il controllo del movimento (Motion Commander) e per i sensori di distanza (Multiranger).

Successivamente vengono specificate le informazioni necessarie per la connessione con il drone e l'inizializzazione dei driver di basso livello.

Tramite le parole chiave "with", a questo punto, si entra in una serie di contesti di gestione che permettono di controllare la connessione con il Crazyflie 2.1, gestire i movimenti ed aver accesso ai dati di distanza forniti dai sensori del Multi-ranger deck.

Nella sezione finale del codice si entra in un ciclo "while" che permette al drone di mantenersi in hovering salvo il verificarsi di una condizione, ovvero quando il parametro "keep_flying" diventa falso. Quest'ultimo scenario può essere indotto dal rilevamento di un oggetto al di sopra del drone ed entro 0.2m da esso, in questo caso il drone atterra. Nel caso, tuttavia, non venga provocato quest'ultimo scenario, il drone mantiene una quota di 0.3m e si muove in direzione opposta agli ostacoli rilevati entro 0.2m da esso tramite i cicli "if" implementati.

2.6 Demo percorso di volo con rilevamento ostacoli

L'ultimo codice presentato (Allegato 3.10) combina un percorso preprogrammato utilizzando alcune funzioni del Motion Commander e dei principi impiegati dal codice implementato dal team di Bitcraze, approfondito nella sezione precedente (2.5).

A questo punto della trattazione è stato sperimentata, infatti, l'implementazione di uno script che potesse verificare la sufficiente ampiezza dello spazio circostante il drone prima di iniziare il percorso preprogrammato.

Nel codice indicato viene riportato come esempio un percorso di volo molto semplice, caratterizzato da un avanzamento di 0.5m, una rotazione di 180 gradi attorno all'asse di imbardata e un ulteriore avanzamento di 0.5m verso il punto di partenza. Gli stessi principi possono però essere utilizzati per qualsiasi percorso di volo a patto di modificare i parametri in gioco.

In questo caso, qualche secondo a seguito del decollo il drone verifica tramite le funzioni "is_close" e "is_closeforward" l'assenza di ostacoli per 0.6m avanti a sé e di 0.2m nelle altre direzioni, mantenuti come margine di sicurezza. Nel caso queste condizioni vengano soddisfatte, il dispositivo prosegue effettuando il tracciato programmato e, a seguito dell'atterraggio, viene riportato nel terminale il messaggio "percorso terminato".

Nell'evenienza, invece, in cui dovesse essere rivelata un'ostruzione al percorso, ovvero ogniqualvolta una fra le funzioni "is_close" e "is_closeforward" riporti come risultato "True", il quadricottero procederà direttamente all'atterraggio, abortendo così il percorso di volo, e verrà stampato nel terminale un messaggio indicante la direzione in cui è stato rilevato l'ostacolo.

Capitolo 3

Sperimentazioni sulla stabilità

In quest'ultimo capitolo verranno brevemente illustrate le caratteristiche di stabilità del drone in caso di pilotaggio manuale o automatico tramite i sensori del flow deck v2.

Per lo svolgimento del suddetto paragrafo è stato utilizzato il Crazyflie PC client, di cui verranno presentate alcune delle proprietà e funzionalità in seguito.

3.1 Crazyflie PC Client

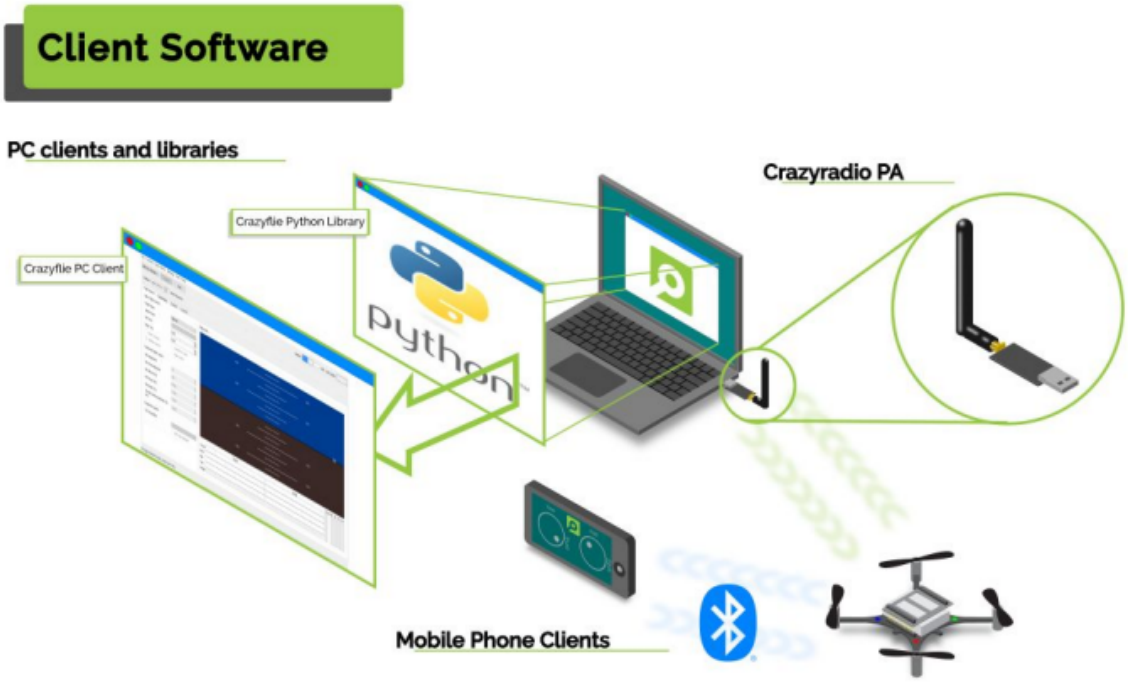


Figura 3.1: Client Software[10]

Il crazyflie PC Client è un software presente nella macchina virtuale fornita da Bitcraze, che permette, in combinazione con la Crazyradio 2.0, la connessione con il drone e la fruizione di diverse praticità. Tra di esse è possibile citare la visualizzazione della telemetria e dei parametri del quadricottero, la possibilità di effettuare un flash del firmware del drone con una versione ufficiale più recente e il collegamento di un dispositivo di input da utilizzare tramite il client stesso, come radiocomando per il pilotaggio del drone.

3.2 Decollo ”manuale”

Come primo test si è verificata la facilità di controllo nel caso di pilotaggio tramite radiocomando e in assenza di alcun deck di espansione attaccato al drone.

Per farlo si è utilizzato un dispositivo di input per il pilotaggio di Crazyflie 2.1 e si è dato tramite una delle levette analogiche il solo comando di spinta propulsiva, senza quindi, a seguito della partenza, effettuare correzioni attorno all'asse di rollio o di beccheggio del quadricottero.

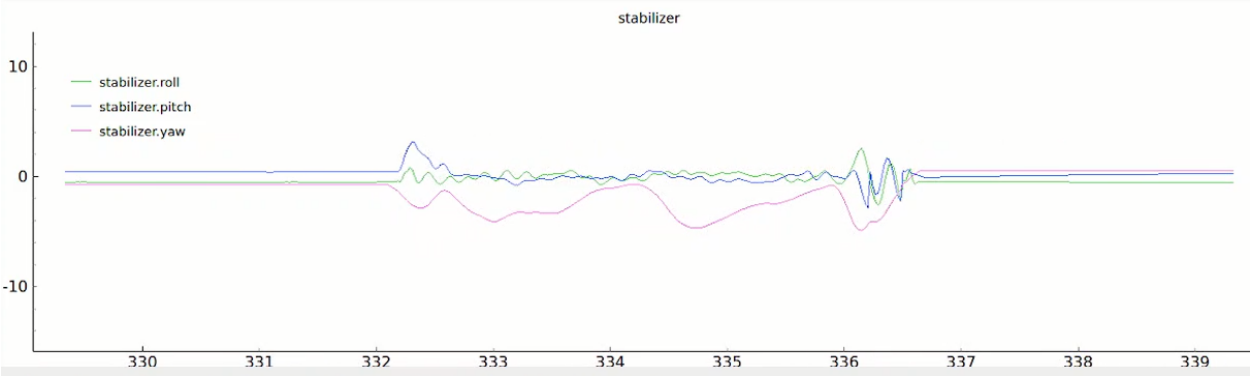


Figura 3.2: Telemetria decollo ”manuale”

In figura 3.2 è riportata la telemetria ottenuta da un decollo, un breve hovering e un successivo atterraggio in pilotaggio manuale.

Com'è possibile notare, fatta eccezione per l'imbardata, probabilmente causata da un non perfetto allineamento della levetta analogica nel radiocomando, il drone risulta avere buona stabilità in rollio e beccheggio. Tuttavia a seguito del decollo, Crazyflie 2.1, risulta soffrire di drifting, in quanto, il dispositivo, sebbene non inclinandosi vistosamente, tende a ”scivolare” verso una direzione. Anche a seguito di trimmaggio dei parametri di beccheggio e rollio, effettuato tramite la corrispondente funzione nel Client, il quadricottero continua a riscontrare questo problema. Probabilmente le cause sono riconducibili a una non perfetta distribuzione della massa e a una spinta propulsiva non omogenea.

3.3 Decollo automatico

Nel caso di decollo automatico, invece, i propulsori intervengono automaticamente per garantire un mantenimento della posizione in fase di hovering.

Per questo test è stato utilizzato il flow deck v2, che permette al quadricottero di rilevare movimenti in x e y.

Come riportato dalla telemetria in figura 3.3, in questa circostanza, si ha inizialmente una alta variazione dei parametri, in particolare quello di beccheggio, corrispondente alla forte spinta propulsiva erogata per effettuare il decollo e necessaria per staccarsi velocemente da terra ed evitare una perdita di controllo del drone causata dall'effetto suolo.

Successivamente ai primi istanti, gli effetti di compensazione ottenuti grazie al flow deck v2 entrano subito in gioco e le variabili di beccheggio,imbardata e rollio, tendono a restare contenute entro certi limiti.

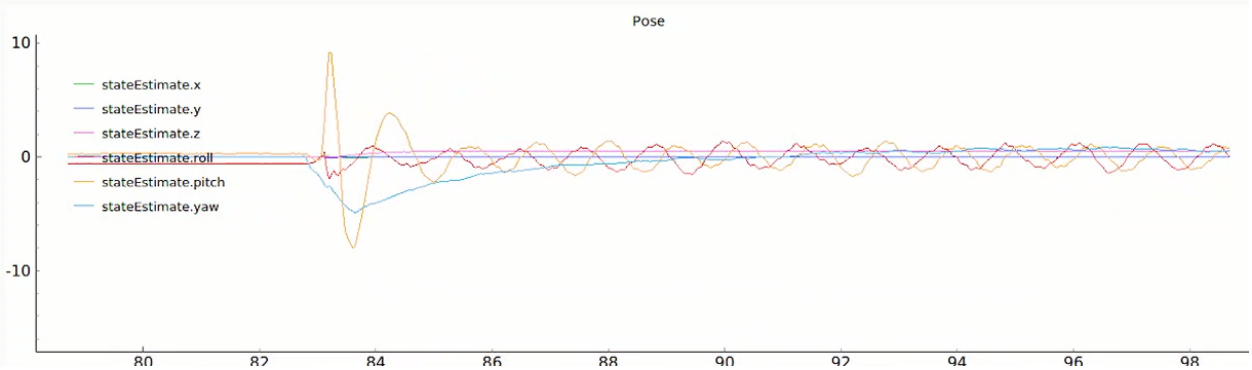


Figura 3.3: Telemetria decollo automatico

In questa modalità, Crazyflie 2.1 mantiene facilmente la posizione in hovering e non si scosta di più di qualche centimetro dalla posizione di decollo originale.

Analisi Risultati e Conclusioni

Come riportato nel corso dell'elaborato, il drone "Crazyflie 2.1" presenta alcune limitazioni, dovute soprattutto alla qualità del dispositivo e alle sue dimensioni, esse vengono però compensate dalla versatilità e le ampie possibilità di sviluppo.

Come sottolineato nel capitolo 3, il quadricottero risulta complesso da pilotare in ambienti ristretti, specialmente da piloti con poca esperienza e scarsa sensibilità nell'utilizzo delle levette analogiche di un radiocomando.

Ulteriori restrizioni conseguono dalla capacità di calcolo computazione limitata, in quanto spesso, specialmente se equipaggiato con due deck di espansione, il drone impiegava diverso tempo per elaborare ed iniziare ad attuare i comandi inviati tramite segnale radio dalla stazione di terra.

Per ultime è possibile citare la bassa autonomia di volo, conseguente dalle dimensioni contenute della batteria e una precisione limitata nel compiere movimentazioni ed azioni richieste tramite codici e funzioni in python.

Questi vincoli vengono tuttavia bilanciati da una varietà di opportunità di sviluppo pressoché illimitata. Sebbene, infatti, la versione "base" del quadricottero non presenti un'ampia dotazione di equipaggiamento e funzionalità, essa è dotata di porte che permettono la connessione di nuovi hardware e piattaforme elettroniche di molteplice natura.

Come evidenziato nel capitolo 2, anche due semplici deck di espansione aprono le porte a una gamma di funzioni molto ampia. In questa ricerca sono state illustrate alcune di queste possibilità, altri codici attuabili avrebbero potuto, a titolo di esempio, riguardare l'utilizzo del multi-ranger deck per mappare la pianta di una stanza non troppo ampia.

La capacità del quadricottero di mantenere una connessione con una stazione di terra, infatti, rende possibile l'impiego del dispositivo anche come sensore per il rilevamento e la registrazione di dati.

Un altro vantaggio di Crazyflie 2.1 è la possibilità di visualizzare e modificare i codici del drone, tra cui il firmware. Quest'ultimo dispone di 1024 Kb di memoria flash nel caso vogliano essere attuate delle variazioni.

In conclusione il drone risulta essere particolarmente adatto per fini accademici e di ricerca. L'utilizzo in ambiente indoor sfruttando le capacità di controllo autonomo è possibile e i settori applicativi possono essere dei più disparati.

Allegati

3.4 Allegato 1

```
import logging
import time

import cflib.crtp
from cflib.crazyflie import Crazyflie
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie

uri = 'radio://0/80/2M/E7E7E7E7E7'

def simple_connect():

    print("Connessione stabilita")
    time.sleep(3)
    print("Disconnessione")

if __name__ == '__main__':
    cflib.crtp.init_drivers()
    with SyncCrazyflie(uri, cf=Crazyflie(rw_cache='./cache')) as scf:
        simple_connect()
```

3.5 Allegato 2

```
import logging
import sys
import time
from threading import Event

import cflib.crtp
from cflib.crazyflie import Crazyflie
from cflib.crazyflie.log import LogConfig
```

```
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander
from cflib.utils import uri_helper

URI = uri_helper.uri_from_env(default='radio://0/80/2M/E7E7E7E7E7')

deck_attached_event = Event()

logging.basicConfig(level=logging.ERROR)

def param_deck_flow(_, value_str):
    value = int(value_str)
    print(value)
    if value:
        deck_attached_event.set()
        print('Deck rilevato')
    else:
        print('Deck non rilevato')

if __name__ == '__main__':
    cflib.crtp.init_drivers()
    with SyncCrazyflie(URI, cf=Crazyflie(rw_cache='./cache')) as scf:
        scf.cf.param.add_update_callback(group='deck', name='bcFlow2',
                                         cb=param_deck_flow)
        time.sleep(1)
```

3.6 Allegato 3

```
import logging
import time

import cflib.crtp
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander

URI = 'radio://0/80/2M/E7E7E7E7E7'

logging.basicConfig(level=logging.ERROR)


if __name__ == '__main__':
```

```
    cflib.crtp.init_drivers(enable_debug_driver=False)

    with SyncCrazyflie(URI) as scf:
        with MotionCommander(scf) as mc:
            print('Decollo')
            time.sleep(3)
```

3.7 Allegato 4

```
import logging
import time

import cflib.crtp
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander

URI = 'radio://0/80/2M/E7E7E7E7E7'

logging.basicConfig(level=logging.ERROR)

if __name__ == '__main__':

    cflib.crtp.init_drivers(enable_debug_driver=False)

    with SyncCrazyflie(URI) as scf:
        with MotionCommander(scf) as mc:
           print('Decollo')
            time.sleep(3)

            print('Movimento in avanti 0.5m')
            mc.forward(0.5)
            time.sleep(2)

            print('Movimento in alto 0.2m')
            mc.up(0.2)
            time.sleep(2)

            print('Arco di circonferenza 90deg');
            mc.circle_right(0.5, velocity=0.5, angle_degrees=90)
            time.sleep(2)

            print('Movimento in basso 0.2m')
```

```
            mc.down(0.2)
            time.sleep(2)

            print('Rotazione 90deg destra')
            mc.turn_right(90)
            time.sleep(2)

            print('Movimento in avanti 1m')
            mc.forward(1)
            time.sleep(2)

            print('Rollio destra 0.5m a 0.6m/s')
            mc.right(0.5, velocity=0.6)
            time.sleep(2)

            print('Atterraggio')
```

3.8 Allegato 5

```
import logging
import sys
import time
from threading import Event

import cflib.crtp
from cflib.crazyflie import Crazyflie
from cflib.crazyflie.log import LogConfig
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander
from cflib.utils import uri_helper

URI = 'radio://0/80/2M/E7E7E7E7E7'

logging.basicConfig(level=logging.ERROR)

position_estimate = [0, 0]

def percorso_volo(scf):
    with MotionCommander(scf) as mc:
            print('Decollo')
            time.sleep(3)
```

```
            print('Movimento in avanti 0.5m')
            mc.forward(0.5)
            time.sleep(2)

            print('Movimento in alto 0.2m')
            mc.up(0.2)
            time.sleep(2)

            print('Arco di circonferenza 90deg');
            mc.circle_right(0.5, velocity=0.5, angle_degrees=90)
            time.sleep(2)

            print('Movimento in basso 0.2m')
            mc.down(0.2)
            time.sleep(2)

            print('Rotazione 90deg destra')
            mc.turn_right(90)
            time.sleep(2)

            print('Movimento in avanti 1m')
            mc.forward(1)
            time.sleep(2)

            print('Rollio destra 0.5m a 0.6m/s')
            mc.right(0.5, velocity=0.6)
            time.sleep(2)

            print('Atterraggio')

def log_pos_callback(timestamp, data, logconf):
    print(data)
    global position_estimate
    position_estimate[0] = data['stateEstimate.x']
    position_estimate[1] = data['stateEstimate.y']

if __name__ == '__main__':
    cflib.crtp.init_drivers(enable_debug_driver=False)

    with SyncCrazyflie(URI) as scf:

        logconf = LogConfig(name='Position', period_in_ms=1000)
```

```
        logconf.add_variable('stateEstimate.x', 'float')
        logconf.add_variable('stateEstimate.y', 'float')
        scf.cf.log.add_config(logconf)
        logconf.data_received_cb.add_callback(log_pos_callback)

        logconf.start()

        percorso_volo(scf)

        logconf.stop()
```

3.9 Allegato 6

```
import logging
import sys
import time

import cflib.crtp
from cflib.crazyflie import Crazyflie
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander
from cflib.utils.multiranger import Multiranger

URI = 'radio://0/80/2M'

if len(sys.argv) > 1:
    URI = sys.argv[1]

logging.basicConfig(level=logging.ERROR)

def is_close(range):
    MIN_DISTANCE = 0.2 #(metri)

    if range is None:
        return False
    else:
        return range < MIN_DISTANCE

if __name__ == '__main__':
    cflib.crtp.init_drivers(enable_debug_driver=False)

    cf = Crazyflie(rw_cache='./cache')
```

```
    with SyncCrazyflie(URI, cf=cf) as scf:
        with MotionCommander(scf) as motion_commander:
            with Multiranger(scf) as multi_ranger:
                keep_flying = True

                while keep_flying:
                    VELOCITY = 0.5
                    velocity_x = 0.0
                    velocity_y = 0.0

                    if is_close(multi_ranger.front):
                        velocity_x -= VELOCITY
                    if is_close(multi_ranger.back):
                        velocity_x += VELOCITY

                    if is_close(multi_ranger.left):
                        velocity_y -= VELOCITY
                    if is_close(multi_ranger.right):
                        velocity_y += VELOCITY

                    if is_close(multi_ranger.up):
                        keep_flying = False

                    motion_commander.start_linear_motion(
                        velocity_x, velocity_y, 0)

                    time.sleep(0.1)

            print('Demo terminata!')
```

3.10 Allegato 7

```
import logging
import sys
import time

import cflib.crtp
from cflib.crazyflie import Crazyflie
from cflib.crazyflie.syncCrazyflie import SyncCrazyflie
from cflib.positioning.motion_commander import MotionCommander
from cflib.utils.multiranger import Multiranger
```

```
URI = 'radio://0/80/2M'

if len(sys.argv) > 1:
    URI = sys.argv[1]

logging.basicConfig(level=logging.ERROR)

def is_close(range):
    MIN_DISTANCE = 0.2  # m

    if range is None:
        return False
    else:
        return range < MIN_DISTANCE

def is_closefoward(range):
    MIN_DISTANCEf = 0.6  # m

    if range is None:
        return False
    else:
        return range < MIN_DISTANCEf

if __name__ == '__main__':
    cflib.crtp.init_drivers(enable_debug_driver=False)

    cf = Crazyflie(rw_cache='./cache')
    with SyncCrazyflie(URI, cf=cf) as scf:
        with MotionCommander(scf) as mc:
            with Multiranger(scf) as multi_ranger:
                keep_flying = True

                while keep_flying:

                    time.sleep(3)

                    if is_closefoward(multi_ranger.front):
                        keep_flying = False
                        print('ostacolo rilevato lato anteriore')
                        break

                    if is_close(multi_ranger.back):
                        keep_flying = False
```

```
        print('ostacolo rilevato lato posteriore')
        break

    if is_close(multi_ranger.left):
        keep_flying = False
        print('ostacolo rilevato lato sinistro')
        break

    if is_close(multi_ranger.right):
        keep_flying = False
        print('ostacolo rilevato lato destro')
        break

    if is_close(multi_ranger.up):
        keep_flying = False
        print('ostacolo rilevato superiormente')
        break

    time.sleep(1)
    mc.forward(0.5)
    time.sleep(1)
    mc.turn_left(180)
    time.sleep(1)
    mc.forward(0.5)
    time.sleep(1)

    keep_flying = False
    print('percorso terminato')
```

Appendice A

Principi di Navigazione e Sensori Inerziali

A.1 Navigazione Inerziale

La navigazione inerziale è una tipologia di navigazione autonoma basata sull'ottenimento della posizione, velocità ed orientamento o assetto del veicolo, rispetto a una condizione iniziale, mediante misure ricavate da sensori inerziali.

L'INS (Inertial Navigation System) è il sistema elettronico che fornisce la soluzione di navigazione ovvero determina le sopracitate informazioni. Esso è costituito generalmente da un IMU (Inertial Measurment Unit), che fornisce delle misure "grezze", e da un computer atto a determinare la soluzione di navigazione partendo dai dati ricevuti in precedenza dall'unità di misura inerziale.

L'IMU, consiste in un dispositivo elettronico, nel quale sono contenute usualmente una terna ortogonale di accelerometri e una di giroscopi da cui è possibile ottenere rispettivamente una misura dell'accelerazione e della velocità angolare.

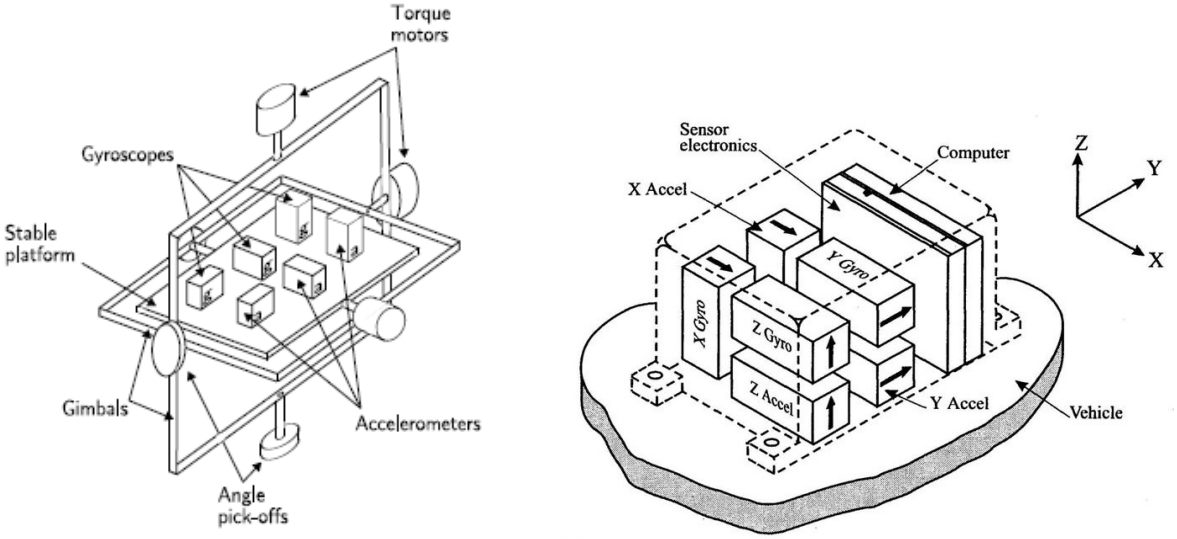
Il sistema di navigazione inerziale si può, infine, presentare in diverse configurazioni, di cui le più comuni sono quella a piattaforma stabilizzante e quella strapdown (fig.A.1a e fig.A.1b.

Nella prima tipologia i sensori inerziali (accelerometri e giroscopi) sono installati su una piattaforma che, grazie all'utilizzo di una sospensione cardanica a tre assi, viene isolata dal movimento esterno del velivolo, mantenendo costantemente l'allineamento a uno specifico sistema di riferimento.

Nella configurazione strapdown, invece, i sensori vengono montati rigidamente direttamente sul velivolo, restituendo così in uscita una misura riferita agli assi corpo del mezzo.

Come è facile intuire, i sistemi strapdown sono meccanicamente più semplici rispetto ai loro corrispettivi a piattaforma stabilizzante, essi necessitano tuttavia di una maggiore potenza di calcolo computazione. Il motivo sta nel fatto che, il computer, per poter risolvere le equazioni di navigazione e fornire quindi la soluzione di navigazione, necessita di accelerazioni e velocità angolari espresse in un sistema di riferimento particolare, chiamato NED. Mentre nei sistemi a piattaforma stabilizzante questo sistema di riferimento risulta

essere quello utilizzato, per gli strapdown non è così, pertanto sarà necessario effettuare una conversione che comporta una maggiore complessità nel calcolo computazionale.



(a) Piattaforma Stabilizzante (b) Strapdown

Figura A.1: Sistemi di navigazione inerziali

A.2 Giroscopi

I giroscopi sono dispositivi usati usualmente per fornire una misura della velocità angolare attorno a un asse di riferimento. Ne esistono di moltissimi tipi, fra cui i principali sono: meccanici, MEMS e ottici.

A.2.1 Giroscopi meccanici

Un giroscopio meccanico convenzionale (fig. A.2) consiste in un rotore montato su due sospensioni cardaniche (gimbals) [19], questo permette la rotazione attorno a tutti e tre gli assi.

Quando posto in rotazione, per la conservazione del momento angolare, l'asse del disco tende a mantenere il suo orientamento. Per questo motivo è possibile sfruttare questa caratteristica e misurare l'angolo tra due gimbals adiacenti per avere un riferimento sulla rotazione subita dal giroscopio.

In questa particolare configurazione il sensore è usato per misurare degli angoli, ma la quasi totalità dei giroscopi moderni fornisce in uscita una stima della velocità angolare.

Sebbene molto semplice come tecnologia e realizzazione, questa tipologia di dispositivo presenta alcune specifiche problematiche, fra cui le dimensioni, in quanto tendono ad essere ingombranti e l'usura, dovuta alle parti meccaniche in movimento, che può causare errori nelle letture e rotture.

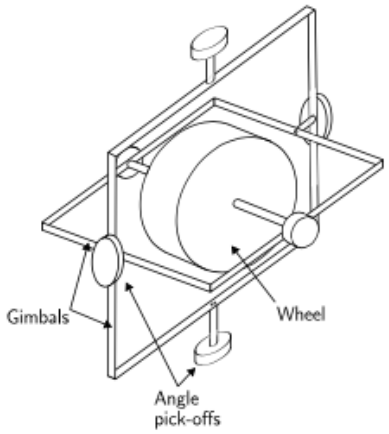


Figura A.2: Giroscopio meccanico tradizionale

A.2.2 Giroscopi ottici

I giroscopi ottici sfruttano la luce per fornire una misura della velocità angolare a cui è sottoposto il sensore. Ne esistono principalmente due categorie: FOG (Fiber-Optic Gyroscope) e RLG (Ring Laser Gyroscope), entrambi si basano sull'Effetto Sagnac, ovvero un fenomeno di interferenza ottica scoperto nel 1913 dall'omonimo fisico francese.

Nel caso del FOG si utilizza una spira di fibra ottica. Dal medesimo punto iniziale vengono emessi due raggi di luce in direzione opposta, che quindi percorreranno la spira in sensi discordi. Se il sistema è posto in rotazione, il raggio di luce che si muove nello stesso senso della rotazione dovrà percorrere un tragitto più lungo rispetto al secondo raggio per l'effetto Sagnac. Questa differenza causa uno sfasamento tra le due onde elettromagnetiche dei due raggi luminosi, che, una volta determinata, fornisce la velocità angolare con cui ruota il giroscopio.

I dispositivi RLG si basano sugli stessi principi dei FOG, a differenza di quest'ultimi, tuttavia, essi indirizzano i raggi laser grazie a un sistema di specchi.

La precisione di questi tipi di giroscopi dipende particolarmente dalla lunghezza del tragitto percorso dalla luce. Maggiore è la distanza, maggiore sarà lo scostamento tra i due raggi e più accurata sarà la misura.

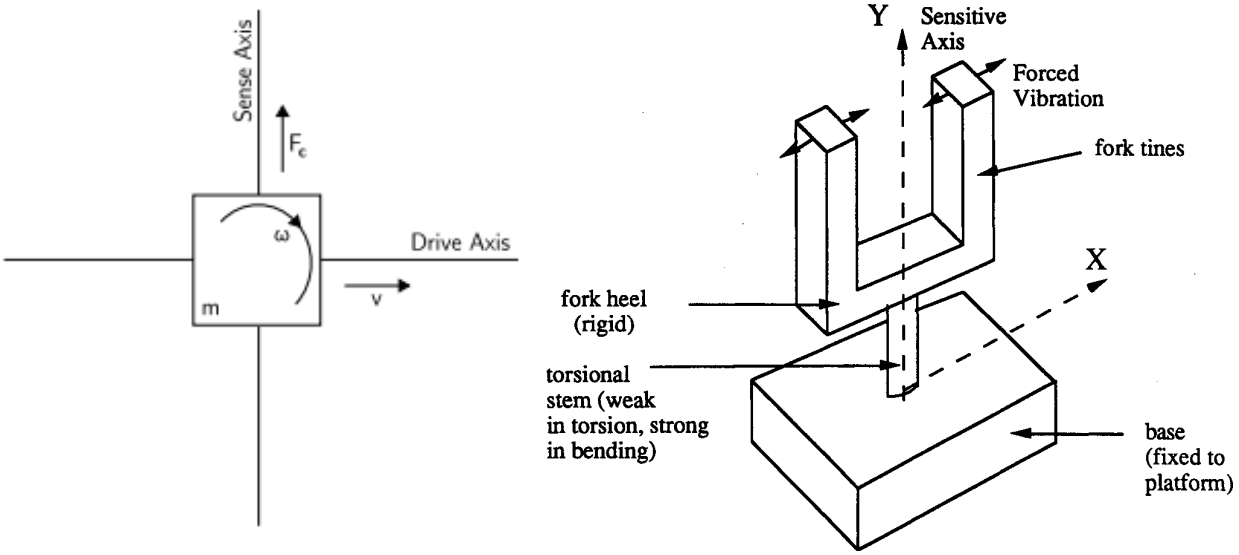
A.2.3 Giroscopi MEMS

I giroscopi MEMS (Micro Electro-Mechanical System) fanno uso dell'effetto di Coriolis, secondo cui un corpo di massa m, che si muove ad una velocità v, in un sistema di riferimento in rotazione rispetto ad uno inerziale con velocità angolare ω, è soggetto ad una forza denominata "di Coriolis" che ha la seguente forma:

$$F_c = -2m(\omega * v)$$

Esistono diversi tipi di configurazioni per i giroscopi MEMS, una di queste, molto semplice, è composta da una massa soggetta a vibrazione lungo un asse [20] (fig. A.3a), quando la massa viene ruotata una seconda vibrazione con asse perpendicolare al precedente sarà introdotta per effetto della forza di Coriolis [21]. Determinando questa seconda vibrazione è possibile risalire alla velocità angolare del corpo.

Un'altra configurazione comune è quella a forchetta riportata in fig. A.3b che sfrutta principi simili alla precedente, ma utilizza come variabile per risalire alla misura di velocità angolare la torsione indotta sulla forchetta stessa.



(a) (b)

Figura A.3: Giroscopi MEMS

Sebbene tendano ad essere meno precisi dei giroscopi ottici, i giroscopi MEMS godono di un massiccio utilizzo, grazie alle loro dimensioni contenute, la facilità di produzione e il basso prezzo oltre che il limitato consumo energetico.

A.3 Accelerometri

Gli accelerometri sono sensori inerziali che forniscono una stima dell'accelerazione lungo una direzione di riferimento. Ne esistono molte tipologie, tra cui è possibile citare: accelerometro meccanico, SAW (Surface Acoustic Wave) e MEMS.

A.3.1 Accelerometri Meccanici

Un classico accelerometro meccanico è usualmente composto da una massa, con possibilità di spostarsi lungo un determinato asse, posizionata in un contenitore ed ancorata ad esso tramite una molla e uno smorzatore. A seguito di un'accelerazione la massa scorrerà in una

direzione scostandosi dalla sua posizione a riposo di una certa quantità. Si può dimostrare, utilizzando la seconda legge di Newton, che questo spostamento della massa è proporzionale all'accelerazione subita da essa.

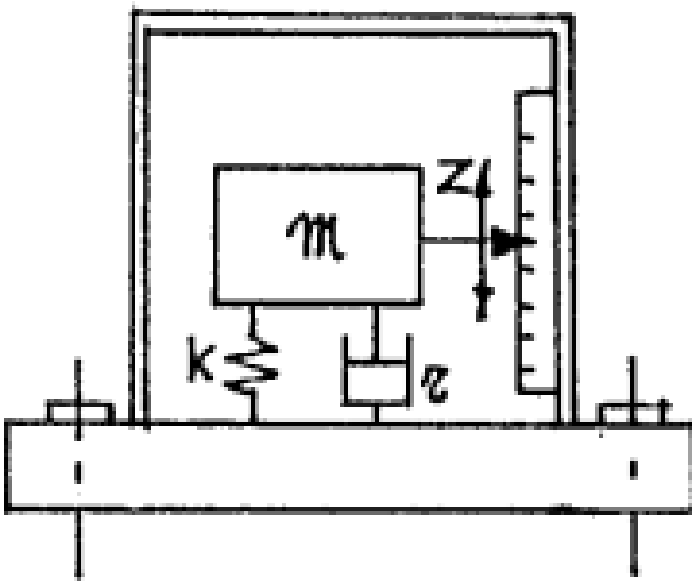


Figura A.4: Schema di funzionamento accelerometro meccanico convenzionale

L'accelerazione misurata da un qualsiasi tipo di accelerometro avrà quindi, in generale, una forma di questo tipo:

$$a_m = a + gsin\theta$$

in cui: a_m è l'accelerazione misurata dal sensore, a è l'accelerazione reale, g è la costante di accelerazione gravitazionale e θ è l'angolo di inclinazione dell'accelerometro (in figura A.4 θ=90 gradi, in quanto disposto in verticale).

Nel caso sia nota l'accelerazione reale a, è quindi possibile utilizzare l'accelerometro come tiltometro o livella, ovvero come strumento per ricavare l'inclinazione θ di un piano.

A.3.2 Accelerometri SAW

Gli accelerometri SAW sono un esempio di accelerometri a stato solido [19],in questo caso si ha una trave a sbalzo attaccata rigidamente al contenitore da una lato e con una massa libera di muoversi dall'altro. La trave inoltre è fatta risuonare a una certa frequenza. Quando un'accelerazione nella direzione preferenziale è applicata, la trave tende a piegarsi, questo porta la frequenza dell'onda acustica di superficie a cambiare di una quantità proporzionale alla tensione meccanica a cui è soggetta la trave. Determinando la variazione di frequenza dell'onda acustica è quindi possibile risalire all'accelerazione subita.

A.3.3 Accelerometri MEMS

Gli accelerometri MEMS si presentano con diverse configurazioni, una di queste prevede una lamina di metallo incernierata all'interno di un foro ricavato in una lamina di silicio. Le due lamine sono separate da aria , che funge da dielettrico.

Dato che la lamina di metallo ha possibilità di inclinarsi in avanti o indietro, è possibile misurare la variazione di capacità elettrica tra metallo e silicio dovuto al fatto che questo spostamento porta un cambiamento nel dielettrico fra esse.

Determinando la variazione della capacità elettrica è quindi possibile risalire alla accelerazione che ha causato l'inclinazione della massa di metallo.

Così come nel caso dei giroscopi MEMS, i vantaggi principali di questa tipologia di accelerometri risiedono nelle dimensioni, nel prezzo e nel consumo energetico limitato.

A.4 Effetto dei disturbi sui sensori inerziali

Come tutti gli strumenti di misura, anche i sensori inerziali tra cui gli accelerometri e i giroscopi sono soggetti a disturbi, i quali causano conseguentemente degli errori sulle letture fornite.

La natura dei disturbi che vanno a inficiare sulle uscite di questi dispositivi è abbastanza disparata, fra essi i principali che è possibile citare sono: rumori bianchi, rumori flickering (flickering noise), disturbi causati dalla temperatura e dalla calibrazione [21].

La principale problematica a cui sono però soggetti accelerometri e giroscopi risiede nel fatto che gli errori a cui sono sottoposti crescono nel tempo nel caso siano necessarie delle operazioni di integrazioni sui valori forniti.

Di fatto, nel caso, per esempio, si voglia ricavare la velocità dalla misura in uscita da un accelerometro, a seguito dell'operazione di integrazione, l'errore risulterà moltiplicato per il tempo, aumentando quindi sempre più durante l'utilizzo del dispositivo.

Questo scostamento dal valore reale, che prende il nome di "bias" per gli accelerometri e "deriva" per i giroscopi richiede l'utilizzo di sistemi, per esempio fonti di misura esterne, per la correzione e l'azzeramento periodico degli errori accumulati.

Appendice B

Principi di Sensori di pressione

Un sensore di pressione è uno strumento in grado di rilevare, determinare e fornire in uscita una misura della pressione a cui è sottoposto.

Nel corso dei secoli sono stati ideati molte tipologie di dispositivi in grado di quantificare questa grandezza, passando da apparecchiature rudimentali e ingombranti ad apparati piccoli e all'avanguardia.

Una prima divisione che è possibile fare tra i misuratori di pressione è quella fra quelli che utilizzano come unica fonte di potenza la pressione (meccanici) e quelli che necessitano di una fonte di potenza elettrica esterna per poter funzionare (elettrici) [22].

B.1 Sensori di pressione meccanici

In questo caso usualmente si ha un componente meccanico la cui forma o posizione viene alterata dalla pressione che sta misurando, in questo modo è possibile far spostare un puntatore in una scala calibrata e avere un'indicazione sulla grandezza in questione. Degli esempi di questi tipi di sensori sono i manometri a U e quelli a membrana.

B.2 Sensori di pressione elettrici

Questo tipo di sensore di pressione, invece, converte la pressione misurata in un appropriato segnale elettrico. Ne esistono una ampia varietà di tipologie distinguibili da una molteplicità di caratteristiche, tra cui il tipo di pressione misurata (assoluta, relativa o differenziale) oppure il principio di funzionamento.

Utilizzando quest'ultimo come criterio di smistamento è possibile catalogare la maggior parte dei sensori di pressione in: piezoresistivi, capacitivi, a fibra ottica, piezoelettrici e a risonanza [23].

B.2.1 Sensori di pressione piezoresisitivi

I sensori di pressione piezoresisitivi sono stati tra i primi prodotti della tecnologia MEMS (Micro Electro-Mechanical Systems), anche per questo motivo sono largamente diffusi ed utilizzati.

Essi si basano sull'effetto piezoresisitivo, ovvero la variazione di resistenza elettrica subita da alcuni materiale, a causa delle sollecitazioni a cui è sottoposto.

Questi tipi di sensori sono usualmente formati da un diaframma, spesso realizzato in silicone per via delle sue buone proprietà elettriche e meccaniche e della facilità di produzione [24], a cui sono integrati dei piezoresistori.

A seguito dell'applicazione di una differenza di pressione, la membrana e i sensori si deformeranno, determinando la conseguente variazione di resistenza elettrica è quindi possibile risalire alla pressione.

I piezoresistori sono usualmente disposti in una configurazione a ponte di Wheatstone e realizzati in materiali metallici o semi conduttivi.

B.2.2 Sensori di pressione capacitivi

Anche in questo caso il sensore genera un segnale elettrico in uscita a seguito di una deformazione subita da una membrana, tuttavia non è lo stress risultante nella membrana a generare il segnale, quanto il suo spostamento [25].

I sensore di pressione capacitivi, infatti, sfruttano la variazione di capacità elettrica risultante dalla variazione di distanza tra le armature di un condensatore, causata dalla pressione applicata, per ricavare quest'ultima.

B.2.3 Sensori di pressione a fibra ottica

In questi tipi di sensori la pressione viene rilevata tramite un effetto sulla luce. Come nei precedenti casi, esistono diversi tipi di configurazioni che possono differire per struttura e funzionamento dagli altri.

A titolo di esempio si prende in considerazione un sensore di pressione estrinseco a fibra ottica basato sull'interferometro di Fabry-Pérot.

In questo caso viene sfruttata l'interferenza che si genera tra più raggi di luce che vengono riflessi tra due superfici, di cui una mobile, in una cavità. La pressione applicata sulla superficie mobile varierà la distanza tra le due aree riflettenti, misurando alcune caratteristiche della luce riflessa è quindi possibile risalire alla pressione applicata.

Questi tipi di sensori offrono diversi vantaggi rispetto alle precedenti tipologie di sensori elettrici, tra cui: un'alta sensibilità e risoluzione ed immunità ad interferenze elettromagnetiche [26]

B.2.4 Sensori di pressione piezoelettrici

Questi tipi di sensori sfruttano l'effetto piezoelettrico. Quando viene applicata una pressione su un materiale piezoelettrico, la deformazione generata da origine a una differenza di potenziale elettrico tra le due superfici.

Determinando questo grandezza è quindi possibile risalire alla pressione.

B.2.5 Sensori di pressione a risonanza

I sensori convenzionali di questo tipo impiegano usualmente degli estensimetri a risonanza per convertire la tensione generata dalla pressione applicata su un diaframma in una variazione di frequenza. Determinare quest'ultima corrisponde quindi a ricavare la pressione a cui era soggetto il diaframma.

L'elemento risonante può essere solidale al diaframma o direttamente integrato in esso per formare un singolo componente.

Il vantaggio principale in questa tipologia di sensori sta nell'alta precisione dato che l'elemento risonante è generalmente molto sensibile.

Sebbene in buona parte dei casi siano realizzati in silicone, anche l'utilizzo di altri materiali e configurazioni è possibile per questi dispositivi[27].

Bibliografia

[1] Treccani, *Drone*, URL: https://www.treccani.it/vocabolario/drone/.

[2] Dji, *Drone Rescues Around the World*, URL: https://enterprise.dji.com/drone-rescue-map/.

[3] Federal Aviation Administration, *Drone by the numbers*, URL: https://www.faa.gov/node/54496.

[4] Bitcraze, *Crazyradio 2.0*, URL: https://www.bitcraze.io/products/crazyradio-2-0/.

[5] Bitcraze, *Crazyflie 2.1*, URL: https://www.bitcraze.io/products/crazyflie-2-1/.

[6] Bosch, *BMI088 6-axis Motion Tracking for High-performance Applications*, URL: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmi088-ds001.pdf.

[7] Bosch, *BMP388 Digital pressure sensor*, URL: https://www.bosch-sensortec.com/media/boschsensortec/downloads/datasheets/bst-bmp388-ds001.pdf.

[8] STMicroelectronics NV, *STM32F405xx STM32F407xx*, URL: https://www.st.com/resource/en/datasheet/stm32f405rg.pdf.

[9] STMicroelectronics NV, *STM32F405xx STM32F407xx*, URL: https://www.st.com/resource/en/datasheet/stm32f405rg.pdf.

[10] Bitcraze, *EPFL presentation 2023*, URL: https://www.bitcraze.io/about/events/documents/EPFL_Bitcraze_lecture_2023.pdf.

[11] Bitcraze, *Flow deck v2*, URL: https://www.bitcraze.io/products/flow-deck-v2/.

[12] STMicroelectronics NV, *VL53L1X*, URL: https://www.st.com/en/imaging-and-photonics-solutions/vl53l1x.html.

[13] PixArt Imaging Inc., *PMW3901MB-TXQT: Optical Motion Tracking Chip*, URL: https://www.codico.com/de/mpattachment/file/download/id/952/.

[14] Bitcraze, *Multi-ranger deck*, URL: https://www.bitcraze.io/products/multi-ranger-deck/.

[15] Bitcraze, *Z-ranger deck v2*, URL: https://www.bitcraze.io/products/z-ranger-deck-v2/.

[16] Bitcraze, *Connecting, logging and parameters*, URL: https://www.bitcraze.io/documentation/repository/crazyflie-lib-python/master/user-guides/sbs_connect_log_param/.

[17] Bitcraze, *Motion Commander*, URL: https://www.bitcraze.io/documentation/repository/crazyflie-lib-python/master/user-guides/sbs_motion_commander/.

[18] Bitcraze, *The STEM ranging bundle*, URL: https://www.bitcraze.io/documentation/tutorials/getting-started-with-stem-ranging-bundle/.

[19] D. Titterton e J. Weston, *Strapdown Inertial Navigation Technology*, IET, 2004.

[20] Daniel Roetenberg, *Inertial and Magnetic Sensing of Human Motion*, PhD thesis, Universiteit Twente, 2006.

[21] Oliver J. Woodman, *An introduction to inertial navigation*, Technical reports published by the University of Cambridge, 2007.

[22] Duane Tandeske, *Pressure sensors selection and applicaztion*, CRC Press, 1990.

[23] Peishuai Song et al., «Recent Progress of Miniature MEMS Pressure Sensors», in: *Micromachines* 11.1 (2020), URL: https://www.mdpi.com/2072-666X/11/1/56.

[24] R. Singh et al., «A silicon piezoresistive pressure sensor», in: *Proceedings First IEEE International Workshop on Electronic Design, Test and Applications '2002*, 2002, pp. 181–184, DOI: 10.1109/DELTA.2002.994611.

[25] Robert Puers, «Capacitive sensors: When and how to use them», in: *Sensors and Actuators A: Physical* 37-38 (1993), Proceedings of Eurosensors VI, pp. 93–105, ISSN: 0924-4247, DOI: https://doi.org/10.1016/0924-4247(93)80019-D, URL: https://www.sciencedirect.com/science/article/pii/092442479380019D.

[26] Yizheng Zhu e Anbo Wang, «Miniature fiber-optic pressure sensor», in: *IEEE Photonics Technology Letters* 17.2 (2005), pp. 447–449, DOI: 10.1109/LPT.2004.839002.

[27] Christopher J. Welham, Julian W. Gardner e John Greenwood, «A laterally driven micromachined resonant pressure sensor», in: *Sensors and Actuators A: Physical* 52.1 (1996), Proceedings of the 8th International Conference on Solid-State Sensors and Actuators Eurosensors IX, pp. 86–91, ISSN: 0924-4247, DOI: https://doi.org/10.1016/0924-4247(96)80130-0, URL: https://www.sciencedirect.com/science/article/pii/0924424796801300.