|  |  |
| --- | --- |
| Verbale n.3 CCdS_LM4_15-03-2018_approvato web | **Corso di laurea magistrale in Ingegneria Informatica**  **A.A. 2021/2022**  **Robotica**  **Docente Prof. Antonio Chella**  **Studenti**  **Luca La Barbera**  **Salvatore Drago**  **Andrea Vaiuso** |
|  |  |

Sommario

[Introduzione 1](#_Toc95087077)

[Idea Proposta 1](#_Toc95087078)

[Generalità del progetto 2](#_Toc95087079)

[Il Robot 3](#_Toc95087080)

[Mavic 2 Pro 3](#_Toc95087081)

[Equipaggiamenti e Sensori 5](#_Toc95087082)

[Supervisor 5](#_Toc95087083)

[Progettazione del sistema 5](#_Toc95087084)

[Requisiti Funzionali 5](#_Toc95087085)

[Requisiti non Funzionali 6](#_Toc95087086)

[Modalità di Programmazione 6](#_Toc95087087)

[Cinematica 6](#_Toc95087088)

[FSDs (Finite States Diagrams) 7](#_Toc95087089)

[Consegna Pacchi 7](#_Toc95087090)

[Contesa Ordini 7](#_Toc95087091)

[Verifica Anomalia 7](#_Toc95087092)

[Implementazione del sistema 7](#_Toc95087093)

[Implementazione controllore 7](#_Toc95087094)

[utils.py 7](#_Toc95087095)

[avoid\_obstacles.py 7](#_Toc95087096)

[score\_calculator.py 8](#_Toc95087097)

[base\_controller.py 9](#_Toc95087098)

[supervisor\_controller.py 9](#_Toc95087099)

[Moto del drone 9](#_Toc95087100)

[Evita ostacoli 11](#_Toc95087101)

# Introduzione

## Idea Proposta

L’idea del progetto nasce da un radicale cambiamento nella società moderna nel modo di fare acquisti.

Oggi le persone tendono spesso ad ordinare la merce tramite le principali piattaforme di e-commerce come per esempio: Amazon, Alibaba, Zalando, etc.

L’incremento esponenziale degli ordini da consegnare e la continua ricerca da parte delle aziende di riuscire a soddisfare il cliente nel minor tempo possibile, portano le stesse a cercare nuovi strumenti che possano velocizzare e automatizzare più lavoro possibile.

La nostra proposta è quella di unire la robotica con il mondo della delivery. Riteniamo che l’utilizzo dei droni per la consegna degli ordini possa essere di grande aiuto per le aziende, diminuendo il carico di consegne per corriere, garantendo una più elevata soddisfazione del cliente.

## Generalità del progetto

L’idea sviluppata è quella di una squadra di droni: Sky Droppers, che gestisce la consegna gli ordini.

L’intenzione del nostro progetto è quella di simulare uno scenario reale in cui gli ordini vengono disposti in delle basi di deposito, dove poi verranno presi e trasportati da un drone per essere consegnati a destinazione. Nel nostro progetto il ruolo di Order Maker è affidato al Supervisor, che dopo aver generato un ordine si occupa di comunicare in broadcast a tutti i droni la base di deposito dove si trova il pacco e l’indirizzo di consegna dello stesso.

Una volta ricevuta la notifica dal supervisor i robot si “contendono” l’ordine per decidere chi dovrà effettuare la consegna. Ciò avviene attraverso un protocollo di comunicazione fra tutti i droni dove ognuno calcola il suo punteggio in base a vari parametri (saranno specificati nella sezione dedicata) e lo invia a tutti gli altri droni. In questo modo ogni drone ha la classifica di tutti i punteggi per ogni ordine ed è in grado di stabilire quali ordini includere nella propria lista di ordini da consegnare.

La consegna di un ordine avviene seguendo un flusso di operazioni che comprendono sia programmazione di azioni che comportamenti del tipo stimolo-risposta.

Ogni drone all’inizio si trova nella propria stazione di ricarica e, dopo avere verificato che lo stato della batteria consenta la consegna di uno o più pacchi che gli sono stati affidati, avvia i motori e si reca nella stazione di deposito associata all’ordine, dove, dopo essersi stabilizzato sopra il pacco, atterra su di esso agganciandolo attraverso un sistema di magnetico. Una volta agganciato il pacco, il drone tramite un sistema GPS e una bussola, contemporaneamente raggiunge l’altezza di navigazione e si pone in direzione della base di consegna. Non appena raggiunta l’altezza di navigazione il drone si avvia verso la base di consegna, dove una volta arrivato si stabilizzerà in posizione per poter diminuire la sua quota fino alla quota di consegna.

Giunto alla quota di consegna il drone sgancerà il pacco che dunque poi sarà ritirato dal cliente (nel nostro mondo simulato il supervisor). Successivamente il drone verificherà lo stato della propria batteria, verificherà se sono presenti degli ordini da consegnare nella propria coda e sceglierà se sia più opportuno andare verso la base di deposito per la consegna di un nuovo ordine o se attendere nella propria stazione di raggiungere un livello di batteria sufficiente prima di ripartire.

Durante tutta l’attività di movimento del drone è attivo un sistema del tipo stimolo-risposta che garantisce che vengano evitati gli ostacoli presenti nel percorso.

Inoltre i processi di consegna degli ordini in coda, contesa di quelli futuri e rilevazione delle anomalie sono in esecuzione parallela e dunque, lo svolgimento di uno dei due macro task non compromette l’altro.

È presente inoltre un sistema di fault-tolerance che assicura, se uno dei droni dovesse subire un guasto e sia dunque impossibilitato ad ultimare la consegna, che un altro drone prenda in carico l’ordine, esattamente dal punto in cui il drone si è guastato.

Attualmente abbiamo simulato tutti i comportamenti descritti precedentemente, tramite il simulatore Webots, modellando un mondo ideale nel quale abbiamo incluso anche degli ostacoli che sono rappresentati da dei muri che si interpongono tra il percorso del drone e la base di consegna.

Per limitazioni nella potenza di calcolo abbiamo dovuto comporre la squadra solamente da due droni, ognuno dei quali ha la propria base di ricarica. Inoltre vi sono un totale di quattro stazioni di consegna e due basi di deposito dei pacchi.

Abbiamo realizzato il nostro progetto avvalendoci del linguaggio: Python 3

# Il Robot

## Mavic 2 Pro

Il Mavic 2 è un drone sviluppato da DJI ed è presente in due versioni:

1. Pro -> quella utilizzata da noi
2. Zoom

Ecco alcuni dati tecnici del robot:

per maggiori dettagli consultare il sito dell’azienda:

<https://www.dji.com/it/mavic-2/info>



.

## Equipaggiamenti e Sensori

Abbiamo equipaggiato il drone con diversi sensori e strumenti utili per la realizzazione dei task:

1. Sei sensori sonar. Rispettivamente uno superiore, uno anteriore, uno a sinistra, uno a destra e infine due nella parte inferiore del robot: uno posizionato nella parte centrale del body del drone che viene sfruttato per il rilevamento della box e delle basi in fase di aggancio e atterraggio, un altro disposto nella parte finale del body slot. I sensori sono stati aggiunti all’interno del body slot aggiungendo per ognuno un nodo “DistanceSensor” ed impostando il type a “sonar”.
2. Una batteria, aggiunta con il nodo “battery”
3. Una bussola, aggiunta con il nodo “Compass”
4. Un Magnete. Agganciato al bodyslot tramite nodo “ Connector”
5. Un dispositivo Emettitore per far si che il robot possa mandare i messaggi utili per la collaborazione con il resto della squadra. All’interno del body slot del drone abbiamo aggiunto un nodo “emitter”
6. Un dispositivo Ricevitore per la ricezione dei messaggi da parte del supervisor e degli altri droni. All’interno del body slot del drone abbiamo aggiunto un nodo “receiver”

## Supervisor

Il Supervisor è un semplice bot che non ha nessuna rappresentazione fisica nel nostro mondo

È stato dotato semplicemente di un emettitore e di un ricevitore per lo scambio dei messaggi con la squadra di droni.

# Progettazione del sistema

## Requisiti Funzionali

1. Notifica ordine
2. Contesa/assegnazione ordine
3. Ricarica batteria
4. Raggiungi base di deposito
5. Aggancia/rilascia pacco
6. Ruota verso base di consegna
7. Raggiungi quota di navigazione
8. Raggiungi base di consegna
9. Ritorna alla stazione di ricarica
10. Evita ostacoli
11. Verifica Anomalia

## Requisiti non Funzionali

1. Un ordine viene preso in carica da un drone in base ad un algoritmo che assegna ad ogni drone un punteggio in base a varie caratteristiche es: posizione, carica della batteria, etc. L’algoritmo elegge un drone vincitore, in maniera tale da garantire che ad effettuare la consegna sarà quello nelle condizioni migliori.
2. Il drone effettua una o più consegne solo dopo avere raggiunto un certo valore di percentuale di batteria, che può variare in base al numero di ordini che deve fare fino al completamento della ricarica, dove in ogni caso il drone, se sono presenti degli ordini nella propria coda, comincia ad effettuare quante più consegne possibili.
3. Divisione del codice in moduli per garantire una buona manutenibilità dello stesso
4. Implementazioni di predicati specifici per il calcolo dei parametri al fine di aumentare la leggibilità del codice.

## Modalità di Programmazione

È stata scelta una modalità di programmazione ibrida: reattiva per quanto riguarda lo svolgimento del comportamento “evita ostacoli”, deliberativa per quanto riguarda lo svolgimento di tutti gli altri compiti.

Nel nostro caso i comportamenti di tipo stimolo-risposta interferiscono continuamente con la parte deliberativa, in quanto è sempre attivo durante ogni fase del moto il controllo dei parametri dei sensori sonar per l’attivazione della funzionalità “evita ostacoli”.

## Cinematica

La cinematica adottata è di tipo inverso, sfruttando la conoscenza della propria posizione tramite GPS, bussola, e la conoscenza a priori delle posizioni delle basi, sia di deposito che di consegna che di ricarica, vengono settate opportunamente le velocità dei motori per il raggiungimento del goal.

I movimenti tipici del drone sono i seguenti:

1. Traslazione verticale
2. Traslazione orizzontale
3. Rotazione attorno l’asse verticale
4. Inclinazione rispetto all’asse orizzontale (fronte/ retro)
5. Inclinazione rispetto all’asse laterale (destra/sinistra)

È opportuno sottolineare come la dinamica di movimento di un drone non comporta l’esecuzione di uno solo dei precedenti tipi di moto per volta ma, anzi, al fine di stabilizzarne la posizione e non causare situazioni che porterebbero lo stesso a danneggiarsi, è corretto, quasi obbligatorio, far sì che, anche se in maniera impercettibile, avvengano dei micro spostamenti in tutte le direzioni del moto.

## FSDs (Finite States Diagrams)

Come anticipato precedentemente, abbiamo realizzato tre macro task principali che vengono eseguiti in parallelo:

1. Consegna Pacchi
2. Contesa Ordini
3. Verifica Anomalia

Di seguito riportiamo i diagrammi di esecuzione dei macro tasks.

### Consegna Pacchi

### Contesa Ordini

### Verifica Anomalia

# Implementazione del sistema

## Implementazione controllore

Abbiamo diviso il codice del controllore in come segue:

All’interno della cartella controllers vi sono due cartelle “base controller” e “supervisor\_controller”

* 1. In “base\_controller” troviamo i files “base\_controller.py” “utils.py” e la cartella “modules”
     1. All’interno di “modules” troviamo i files “avoid\_obstacles.py” e “score\_calculator”
  2. In “supervisor\_controller” troviamo il file “supervisor\_controller.py”

Proseguiremo il paragrafo descrivendo le funzioni implementate nei vari files commentando opportunamente lo scopo di queste nel funzionamento globale del controllore

### utils.py

In questo script si trova l’implementazione del codice di due classi:

1. StabilizationStack
2. Coordinate: utile per settare e ritornare le coordinate di un drone o oggetto nel mondo

E due metodi:

1. *Euc\_dist()*: che restituisce la distanza euclidea fra due punti dati in iput
2. *getID()*: che dato il nome di un drone ne restituisce l’id

### avoid\_obstacles.py

Questo script viene utilizzato per l’attivazione e la gestione del comportamento “evita\_ostacoli”.

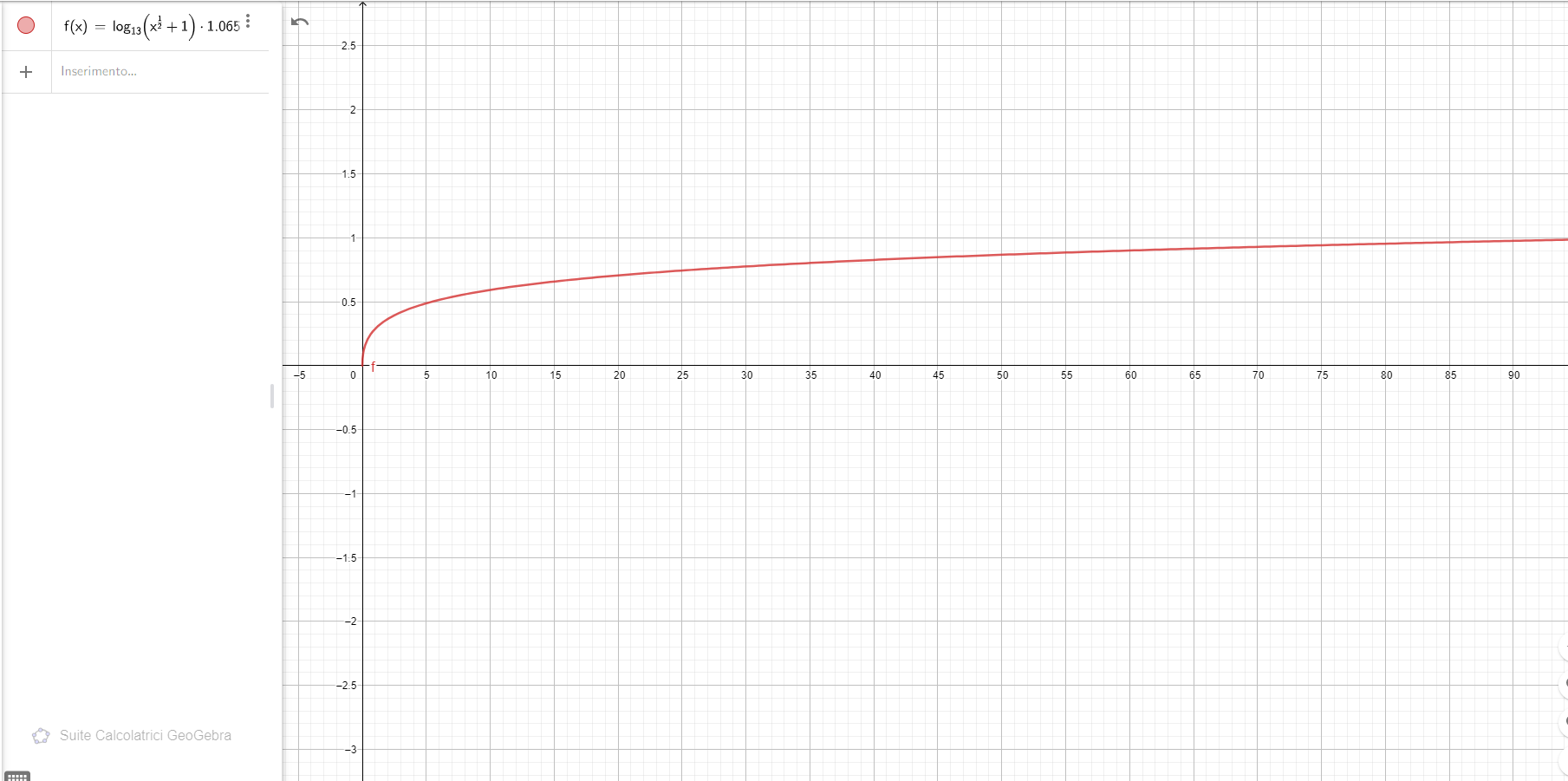
Abbiamo scelto di modellare l’attivazione del comportamento evita ostacoli con questo modulo in cui la funzione “*function(x)”* modella la sensibilità della soglia di attivazione dello stato “avoid\_obstacles”. Ciò viene modellato con una funzione logaritmo, poiché riteniamo che essa approssimi in maniera ottimale e naturale il comportamento di una soglia di attivazione che: è equivalente al valore massimo di distanza che può rilavare il sonar quando il drone viaggia a della velocità elevate, mentre si abbassa proporzionalmente con la velocità in condizioni in cui il drone si trova a navigare a velocità medio/basse.

Con la funzione *avoid\_obstacles\_sensor(value, velocity),* attiviamo o meno la rilevazione da parte di uno dei sonar di un ostacolo.

La funzione *avoid\_obstacles\_full(upper\_sensor, front\_sensor,left\_sensor, right\_sensor, velocity)* ritorna “true” quando uno dei sensori rileva un ostacolo, “false” altrimenti. Viene utilizzata come funzione di controllo in tutti gli stati di movimento, per l’attivazione del comportamento “evita ostacoli”.

In basso viene mostrato il grafico della funzione logaritmo che abbiamo scelto :





### score\_calculator.py

Questo script è stato scritto per l’implementazione del calcolo dello score di un drone rispetto ad un ordine.

Questo score esprime una stima in secondi del tempo necessario al drone per ultimare l’ordine corrente, tutti i suoi ordini in coda e in aggiunta quello per cui si sta calcolando lo score.

La funzione sccalc() prende in input la lista degli ordini in coda, l’ordine per cui si sta calcolando il punteggio(pending), l’ordine corrente, la storia degli stati, la posizione attuale del robot e le coordinate della propria stazione di ricarica. Il calcolo viene effettuato come segue

Per tutti gli ordini da fare e quello preso in esame(pending), viene calcolata la distanza di andata e ritorno, tale valore viene successivamente moltiplicato per una costante K=1.7, in quanto il robot percorre in media 1m in 1.7s. Sono poi aggiunte delle costanti empiriche, valutate sperimentalmente, come:

1. tempo medio di decollo
2. tempo medio di atterraggio
3. tempo medio di aggancio/rilascio box
4. tempo medio impiegato nell’evitare ostacoli lungo un tragitto

Una procedura simile viene effettuata per il calcolo del tempo che serve per concludere quello corrente. Tale tempo viene calcolato in maniera differente in base alla storia degli stati e alla posizione corrente.

### base\_controller.py

Questo è il file principale per il controllore del nostro drone, dove vengono implementati quasi tutti gli stati dei diagrammi visti in precedenza. Il file è diviso in due parti principali:

1. la prima: dove vengono istanziate tutte le variabili globali di ambiente e vengono implementati gli algoritmi di calcolo dei parametri per il moto del drone, vengono anche implementati i thread che gestiscono la comunicazione con la squadra di droni e il supervisor
2. la seconda: dove viene implementato il diagramma a stati che riguarda la consegna dei pacchi e la verifica di una anomalia.

### supervisor\_controller.py

Questo è il modulo che si occupa del robot supervisore, che nel nostro mondo ha il ruolo di order maker.

## Moto del drone

Il moto del drone viene gestito con un sistema ibrido deliberativo/reattivo. La pianificazione dei vari goal da raggiungere viene fatta a priori e, la verifica del raggiungimento degli stessi, al fine di passare da uno stato all’altro, è affidata al controllo dei valori dei dispositivi di cui è fornito il drone stesso.

A differenza di altre situazioni, la gestione del moto di un drone richiede una diversa interpretazione dei concetti di stazionarietà e posizione. Infatti, tranne nel caso in cui il robot sia fermo nella stazione di ricarica, non si avrà mai il raggiungimento preciso di una posizione esatta, ma piuttosto bisogna considerare delle posizioni che sono degli intorni di approssimazione sui quali il robot effettua una continua stabilizzazione, calcolando una certa quantità di errore nel considerare raggiunto un determinato goal.

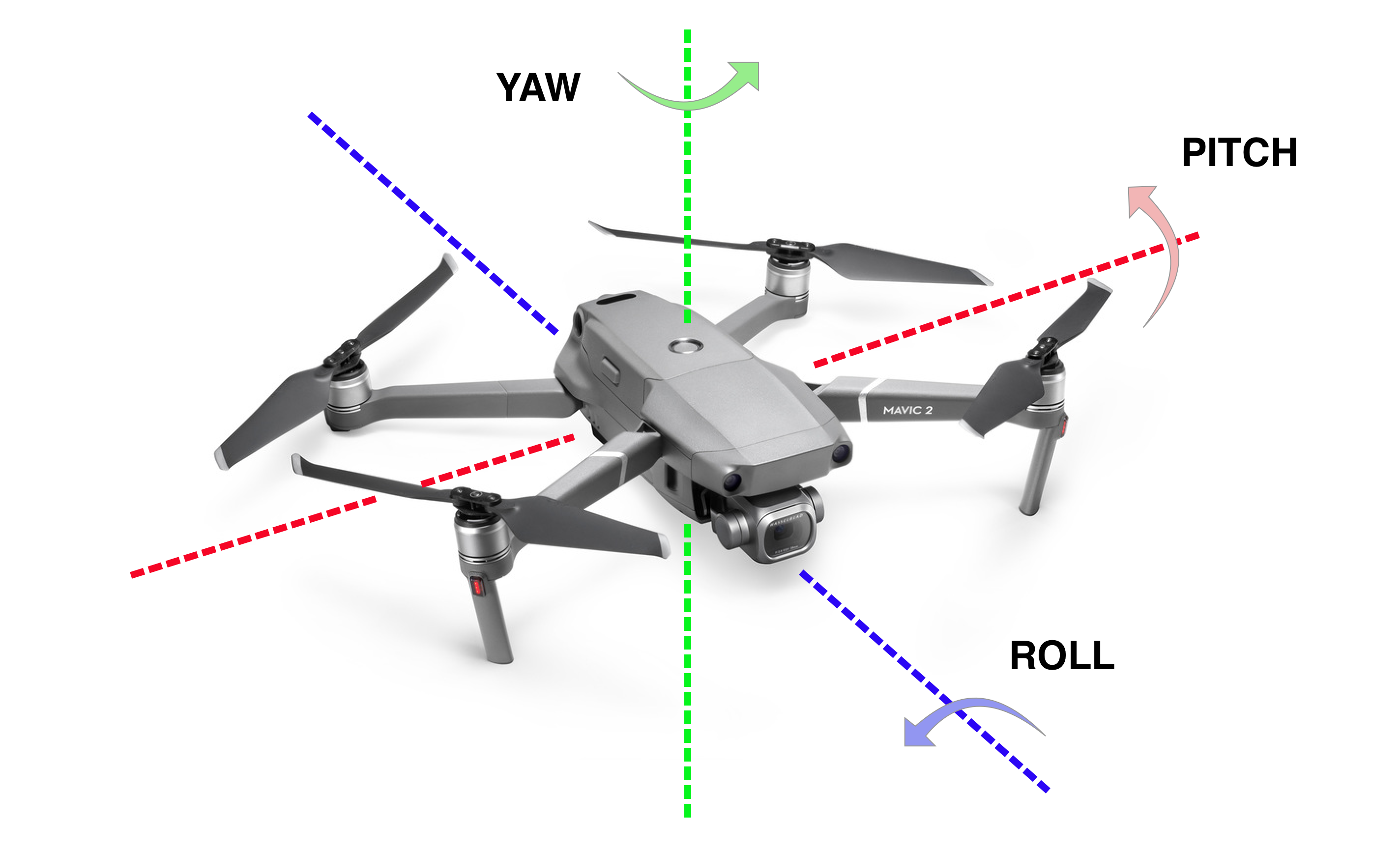
L’obiettivo che ci siamo posti è quello di far svolgere tutti i movimenti al drone in totale sicurezza, modellando uno scenario in cui i percorsi per raggiungere i goal prevedono come tipi di movimento:

1. il raggiungimento della quota prevista
2. la correzione dell’angolo rispetto al quale è rivolto il fronte del drone, allineandolo in direzione del goal
3. il raggiungimento in linea d’aria del goal
4. la stabilizzazione intorno alla posizione del goal

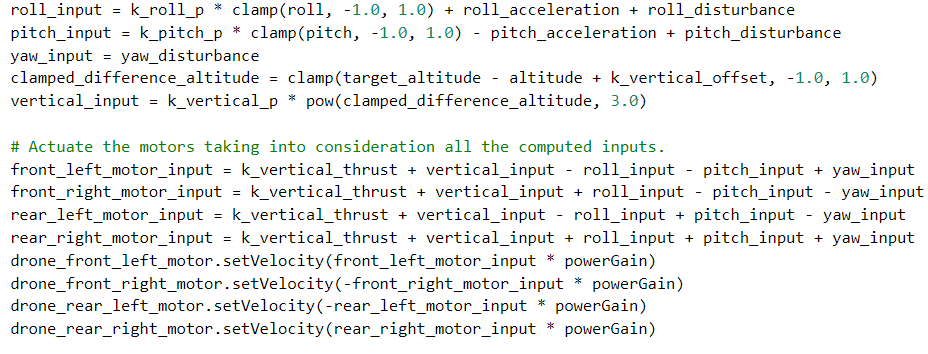
Per evitare il caso in cui un drone dovesse perdere di vista il goal e andare in un’altra direzione, il controllo per la stabilizzazione dell’angolo viene fatto di continuo durante il tragitto. L’effetto ottenuto è quello di un indirizzamento quasi perfetto nella direzione del goal.

Il drone è fornito di 4 rotori: 2 anteriori e 2 posteriori. La velocità dei quattro rotori regola il livello di imbardata, rollio e beccheggio. Il controllore agisce su un algoritmo di stabilizzazione che costringe il robot a mantenere la posizione. Il movimento è controllato dall’alterazione delle variabili:

1. Yaw\_disturbance
2. Roll\_disturbance
3. Pitch\_disturbance
4. Target\_altitude



La modifica dei valori di queste variabili influenza in maniera lineare le velocità dei 4 motori che vengono calcolate come segue:



Oltre alle variabili che abbiamo dichiarato sopra, abbiamo introdotto una costante globale “powerGain”, che ci permette di gestire direttamente a nostro piacimento la velocità dei 4 motori.

### Evita ostacoli

L’implementazione del comportamento evita ostacoli è stato realizzato interamente come un sistema del tipo stimolo-risposta. Durante il movimento del drone, in ogni stato, vi è il controllo sui valori dei sonar e, qualora uno dei sensori dovesse scendere sotto il livello di soglia, la gestione del moto del drone passa allo stato “avoid\_obstacles”.

Nel nostro mondo gli ostacoli possono essere di due tipi: mobili e statici. Gli ostacoli statici nella simulazione sono rappresentati dai muri che vengo frapposti tra il percorso del robot e la base di consegna, mentre quelli dinamici per semplicità sono rappresentati dei droni stessi che durante il volo potrebbero avere delle traiettorie che si intersecano

In questo stato il comportamento implementato è quello di “allontanarsi” nella direzione opposta rispetto alla quale è stato rilevato l’ostacolo. Infatti l’implementazione consiste nella modifica delle 4 variabili: target\_altitude, yaw\_disturbance, roll\_disturbance, pitch\_disturbance.

La scelta di quale variabile modificare avviene attraverso il riconoscimento di quale sia il sensore che attiva in comportamento per evitare gli ostacoli. L’attivazione da parte di uno o più sensori fa si che vengano modificate contemporanemente 1 o più delle 4 variabili che gestiscono il moto del drone.

È possibile che l’ostacolo risulti rilevato, per esempio, sia dal sensore di sinistra, che da quello frontale, in questo caso vengono modificate entrambe le variabili, rispettivamente roll\_disturbance e pitch\_distrubance

Il controllo del moto ritorna alla fase deliberativa solo quando tutti i valori dei sonar risultino non rilavare nessun ostacolo.